Modern Robot Engineering

Edited by E. POPOV

Advances in Science and Technology in the USSR

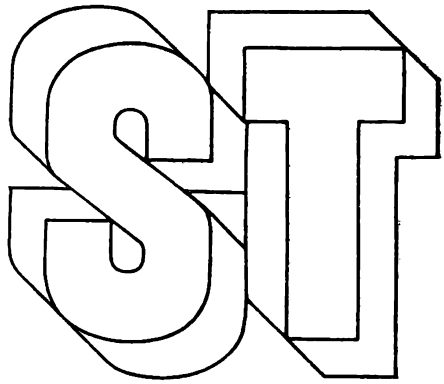
Technology Series

MIR Publishers
Moscow

Mir Publishers Moscow

Advances
in
Science
and
Technology
in
the USSR

Technology
Series



Modern Robot Engineering

Edited by E. P. POPOV,
Corr. Mem. USSR Acad. Sc.

Translated from the Russian by
Grigory Pasechnik

MIR Publishers
Moscow

Современные проблемы робототехники
Под редакцией чл.-корр. АН СССР
Е. П. Попова

First published 1982

На английском языке

© Издательство «Мир», 1982
© English translation, Mir Publishers, 1982

Printed in the Union of Soviet Socialist Republics

Contents

Foreword 7

Preface 9

1 **A Mathematical Model of a Manipulator-served Production Flow Line,** *I. M. Makarov* 13

2 **Mathematical Models of Robot Dynamics,** *V. S. Medvedev* 23

2.1 General 23
2.2 Coordinate Systems 23
2.3 Basic Kinematic Equations 31
2.4 Partitioned Matrix Representation of Main Kinematic Relationships 37
2.5 Dynamic Equations of Robot Arm Mechanism 42
2.6 Dynamic Equations for a Moving Base Robot 48
2.7 Linearizing Equations of Motion 51

3 **A Frequency-domain Approach to Synthesis of Robot Actuator System,** *V. S. Medvedev and L. P. Shigorin* 62

3.1 General 62
3.2 Spectral Frequency Response Analysis 65
3.3 Frequency Responses of Robot Actuator Control System 72
3.4 Synthesis of Robot Actuators with Spectral Frequency Responses 75

4 **Path Synthesis for Robot Linkage in Crowded Workspace,** *V. L. Generozov* 82

4.1 General 82
4.2 Modeling Arm Linkage and Environmental Workspace 82
4.3 Evaluating Distances and Their Subdifferentials 86
4.4 Software Realization and Numerical Modeling 90

5 **Modeling Robot Control in Assembly Operations,** *F. M. Kulakov* 100

5.1 General 100
5.2 Mathematical Model 100
5.3 Possible Approaches to Problem of Control 105
5.4 Minimizing Objective Functional Over Position Vector Domain 106
5.5 Minimizing Objective Functional Over Generalized Coordinate Vector Domain 109
5.6 Extrapolating Programmed Path 112
5.7 Method of Control Cutting Amount of Computation 113

6 **Robot-oriented Software for Computer-assisted Teaching and Control,** *S. L. Zenkevich and A. V. Nazarova* **116**

6.1 General 116
6.2 Design of Robotic Complex and its Operating System 116
6.3 Robot Motion Control Language 126
6.4 Motion Control Software 131
6.5 Teaching of Robot 133
6.6 Testing of Communication Channels 185
6.7 Computer Graphics 136
6.8 Conclusion 138

7 **Methods for Evaluating Robots' Software,** *P. I. Rudakov* **139**

7.1 General 139
7.2 Pattern Recognition for Evaluating Software 139
7.3 Game Theory Analysis of Software Performance 144

8 **Semiautomatic Robot Control Systems,** *A. S. Yushchenko* **149**

8.1 General 149
8.2 Semiautomatic Control Algorithms 150
8.3 System of Manipulator Hand Self-directing Control 156
8.4 Dynamic Control of Manipulators 159
8.5 Analysis of Semiautomatic Control Systems 165

9 **Synthesis of Walking Motion Control,** *D. E. Okhotsimsky, A. K. Platonov, E. A. Devyanin, and V. S. Gurfinkel* **176**

9.1 General 176
9.2 Walking Vehicle as a Complicated Robotic System 176
9.3 Simulation of Control Algorithms 178
9.4 Physical Modeling of Walking Vehicles 184
9.5 Results 189
References 192

Foreword

Integrated automation in industry entails wide application of robotic systems. They may comprise manipulators built into production lines and capable of performing simple preprogrammed operations controlled by the same minicomputer which commands the technological flow processes. More sophisticated production may be automated with robotic manipulators which can be programmed individually or be controlled by a master computer. With individually controlled robots the control is furnished via a microprocessor, whereas the control system of the group-controlled manipulators consists of a few microprocessors, dispatched by a master computer. Thus, robots may interchange their functions noninterruptionally should the technological process vary, repair and maintenance be needed, or emergency occur.

As the experience on the existing robots has accumulated, the need arose for additional capabilities. Among them are provision of robots with rudimentary sensory perception, development of supervisory controlled robots, which can be easily readjusted while going from one task to another, creation of artificial intelligence robots capable of performing operations under conditions of uncertainty, synthesis of algorithms to guide the robot around unknown obstacles, and computer-aided interaction with and design and manufacture of robots.

The manipulator adjustment can be drastically improved by invoking a digital computer which organizes robot self-teaching and changes its working programs automatically. This calls for the development of computer assisted methods of modeling the manufacturing processes. The objective of this modeling is to simulate for a robot certain standard and emergency conditions. It is advantageous in that it greatly reduces the time of putting robotic systems into operation. However, since modeling should be done in real time scale it is difficult to reproduce high rate production processes. Another aspect is robot reliability. The hardware and software of robot control should be devised so as to ensure that the operation of microprocessors is reliable and immune to noise. The operation in aggressive media harmfully affects the actuating and metering devices and reduces the service life of robotic systems. All this greatly complicates the design and manufacture of automated complexes.

Many research institutions in this country successfully solve many of these problems. This allows automatic control systems to be developed and built into production lines of metal-working, mining, and chemical industries. In the years 1976-1980 more than 8 000 systems were put into operation, increasing the robot population by 7 000 machines.

The years 1981-1985 will see a threefold increase in production of built-in automatic systems and robots. This will contribute to gains in productivity, improve quality, reduce costs and, last but not least, relieve man of tedious, dangerous and otherwise unpleasant jobs.

This book is a collection of papers by leading Soviet authorities on robotics cemented by a unified approach to the problem of robot kinematics, path synthesis, and control. The papers consider the selection of structure and parameters for robotic systems and robot served flow lines, analyze the dynamics and kinematics of robot arm linkage, and synthesize paths in crowded workspace. Among the topics of the book are programs specifically devised to control and teach robots, along with the comparison of these programs for performance.

The presented theory is discussed as applied to built-in robotic systems, robots with various principles of control, and walking vehicles.

The Authors hope that the book will acquaint the readers with a number of problems in robotics being solved in the Soviet Union.

E. P. Popov

Preface

Industrial robots find ever increasing application in instrument making and machine building. Gradually robotic systems penetrate into other branches of the national economy. Robots and robotic systems relieve man from mindless drudgery of performing tedious, repetitive and dangerous jobs.

At present many people are engaged in manual labour not only because of insufficient development of mechanization and automation but also because a number of operations cannot be performed without participation of man. In a variety of industrial operations the operator guides multiarticulation manipulator and the slave follows it in point-to-point geometric isomorphism.

The development and wide application of robots integrated with automatic industrial equipment is not only a technological and economic but also an important social objective. Manpower deffi-ciency is anticipated for many jobs requiring unskilled labor. On the one hand, it is due to a general rise in the educational and cultural level of youth and, on the other, to the unfavorable demographic prognosis.

Social progress necessitates significant reduction of unpleasant and dangerous manual jobs leaving a man only the creative aspects of work. Moreover, the development of robotics helps overcome the difference between mental and physical labor.

A wide use of automatic manipulators and robot-assisted complexes yields positive economic effects for output rate, product quality, adaptability to tools changing, quick readjustment, etc.

Industrial robots consist of two major component systems. First, there are the moving parts chiefly comprising the arm, wrist, and hand elements, actuated by drive mechanisms. Complementary to the moving robot system is the control system, exercising control over limb drives and giving the robot a programmable memory.

The robotics embraces an experience of constructing numerically controlled machine tools and servo systems with a great range of speed regulation.

A multiarticulation mechanical arm with controlled servos allows the robot to perform a wide range of different jobs with various workpieces, parts, or tools. What differs the robot from conventional

automations is its ability to operate on its own which makes possible, first, to unman many operations and, second, to quickly readjust from one job to another while modernizing the technological process or shifting to a new type of production.

During the last decade the development of microelectronics, advances in computer technology, and the availability of reliable electromechanical and hydromechanical servo mechanisms have contributed to robot design.

The sphere of human activities comprises, besides in one or another way unpleasant work, a multitude of dull repetitive jobs to be found in design, drawing, metering, and library work. Automation advance will lead to a new class of robots relieving man from tedious labour. As distinct from usual manipulators these data-controlling robots will be furnished with specially designed end effectors.

Robots in technology both manipulators and computer-augmented are a far cry from the robots of popular fiction, to say nothing of man. Nevertheless robots can outperform man in hostile working environments, or while doing tedious repetitive or otherwise unpleasant jobs.

Work configuration for a robot should be organized so as to establish optimal working layout.

Besides the industrial robots substituting human machine operator, automation necessitates the design of computer-augmented robots, which will not only perform the control and synchronizing functions, but also rationalize data flow through the factory.

Another class of robots represent moving or mobile robots. These may be wheeled, tracked, or legged machines. They can find wide application in shops, warehouses, railway stations, on and off roads, in the air and water. Control is furnished automatically.

The development of automatically controlled industrial robots has witnessed three generations: programmable, supervisory controlled, and artificial intelligence robots.

The term "generation" is used here quite conditionally. Robots generations do not change each other historically. New robots generations come to life, while the first generation continues to live and finds ever increasing application. Robot design is perfected and performance improved. Different types of robots of each generation find application in industry.

The first generation, i.e. programmable robots go steadily through their programmed movements. To illustrate, the robot picks up a workpiece, loads the press, unloads the press, moves the workpiece to another place, returns to the initial position. For this purpose use is often made of a two-arm robot (Fig. 1.1). Then picking up a workpiece by one hand coincides with unloading the press by another, next one hand loads the press and another moves the workpiece to another place, while the press operates, both hands return to the

initial position and the operations repeat.

A programmable robot is easily readjusted to various working programs. But once adjusted it is capable of performing only programmed motions.

The first generation comprises industrial robots widely used in the late 70s. They will remain as a basic type for the forthcoming five years but at the same time supervisory controlled industrial robots are created.

The second generation, supervisory controlled robots, are provided with rudimentary sensory perception: tactile, force, locational, visual and heat. Since it is possible to send and receive signals to and from the associated equipment, the robot has the ability to control operation depending upon the received information. Thus, it has to possess certain adaptivity, which is essential in motion, operations, involving parts with undetermined orientation, assembly, welding, etc.

The third generation, artificial intelligence robots, possess richer means of sensory perception, pattern recognition, decision making, and motion implementation. Visual perception, digital simulation of the environment and complex use of sensors assume herein a special significance.

One must bear in mind that the term "artificial intelligence" like terms "computer memory", "technical vision", and others widely used nowaday reflects only external resemblence to man's functioning and bears no analogy as to design and operation principles. Contemporary engineering does not attempt to imitate human prototype but relies on its own design principles. At the same time, roboticists show great interest in bionics designing systems modeled after living organisms.

The contemporary state of engineering knowledge and practice limit the development of artificial intelligence robots. The advancement of science and production improvement will obviously lift the limitations and consequently widen robot capabilities.

For designing supervisory controlled robots and, in particular, artificial intelligence robots the greatest importance present: first, creation of means of sensory perception; second, miniaturization of computer means (built-in microprocessors); and, third, development of software insuring real time data processing. The latter requirement necessitates that only a necessary minimum of information be sensed. This especially concerns visual sensors which give excessive information on the state of environment.

The classification of industrial robots by the generations relies on the basic principles of designing their control systems.

Moreover, industrial robots may be classified into categories according to robot drives (hydraulic, pneumatic, electric), number of joints in the manipulator (from three to eight), and load capacity (from grams to hundreds of kilograms), etc.

Industrial robots are also classified according to the field of their application to machine building, instrument making and to other branches of national economy.

As the technological process shows it is important not only to create a robot, but also to connect it with other equipment. In this way robot-assisted technological complexes appear. They differ in composition depending on the type of the robot application: metal-working, forging, casting, press-work, welding, painting, assembly, etc. However, the robots proper being multipurpose machines may be of a similar type in different complexes. Herein mention should be made of the manipulators controlled by the human hand, balanced loading manipulator including. Having hooked on a load, the operator moves the load in a required direction manipulating a control handle and thus generating control signals to the manipulator actuators.

Man has expanded his horizons beyond his natural environment, first, by devising protective means which limit his capabilities in performing different jobs and, second, by designing machines for operation in extreme conditions.

Physiologically hostile environments that include radiation, gas-contamination, strong microwave radiation, strong magnetic fields, excessive heat or intense cold, high vacuum, e.g. in outer space, great pressures, e.g. in deep water, require that man be completely excluded from operation. This contributed to the elevation of a special branch of robotics, i.e. development of computer-augmented robots. They are used for the purpose of exploring and surveying the environment, and also for controlling joint operation of robots.

Master-slave manipulator systems have long been used for performing jobs in extreme conditions. The operator guides the model of the manipulator and the remote slave follows it in point-to-point geometric isomorphism, or the operator uses separate switches, or a joystick, to specify the direction and speed with which the manipulator arm moves.

In a supervisory controlled system as distinct from semiautomatic systems and especially master-slave manipulator systems the human operator is relieved of a tedious task to perform all operations by his hand. He has only to supervize the system operation and specify task subgoals to a remote computer.

With further development of supervisory control the operator will define not task subgoals but rather goal-oriented program. Besides that supervisory control relieves man of tedious-to-perform manual operations, which leads to increased productivity at the robotized work stations.

The use of robots in national economy should be evaluated not only on a cost/benefit but also an a social improvement basis, which becomes prevailing in the elimination of debilitating jobs.

1 A Mathematical Model of a Manipulator-served Production Flow Line

I. M. Makarov, *Corr. Mem., USSR Acad. Sc.*

Integrated automation in industry largely depends upon the versatility of both individually programmed and master computer controlled robotic manipulators.

Individually controlled robots are employed in those automated systems where technological processes are organized by means of separate flow lines. The manipulators serving such a line need not be controlled by a master computer since they perform on an individual basis. A workpiece goes from one machining tool to the next upon completion of a sequential technological operation. The completion of one operation signals the start for the next so that generally a single micro- or mini-computer may be entrusted with the control of such flow lines. These robots are most popular in manufacturing systems built around CNC machines. They can also be employed on flow lines manufacturing printing circuits or simple chemicals. Figure 1.1 exemplifies robots in an automated flow line having a CNC machining tool.

Group-controlled robots operate on the interrelated processing systems such as assembly lines, refinery technologies, or combined manufacture (molding, machining, assembling, and painting). The control system generally consists of a few micro-processors dispatched by a master computer. Group-controlled manipulators may also be involved when a few parallel flow lines interchange the processed items. The workflow can be designed so that the robots may interchange their functions non-interruptionally should the flow pattern be changed or technological conditions varied, thus providing for more flexibility and redundant reliability for the entire system.

In what follows we would like to consider a few examples of automated manufacturing processes serviced by robots, which are individually or master-computer controlled, and to generate mathematical models which describe their behavior from the viewpoint of waiting line theory.

In an automatic flow line, robots load workpieces to and unload them from machining tools, transfer workpieces from one machine to the next, gauge the machined parts, and sort them into store cells. Also they change cutting tools should these break down or wear out, collect chips to bale them and transport to a disposal store. All these robots belong to the variety of computer augmented

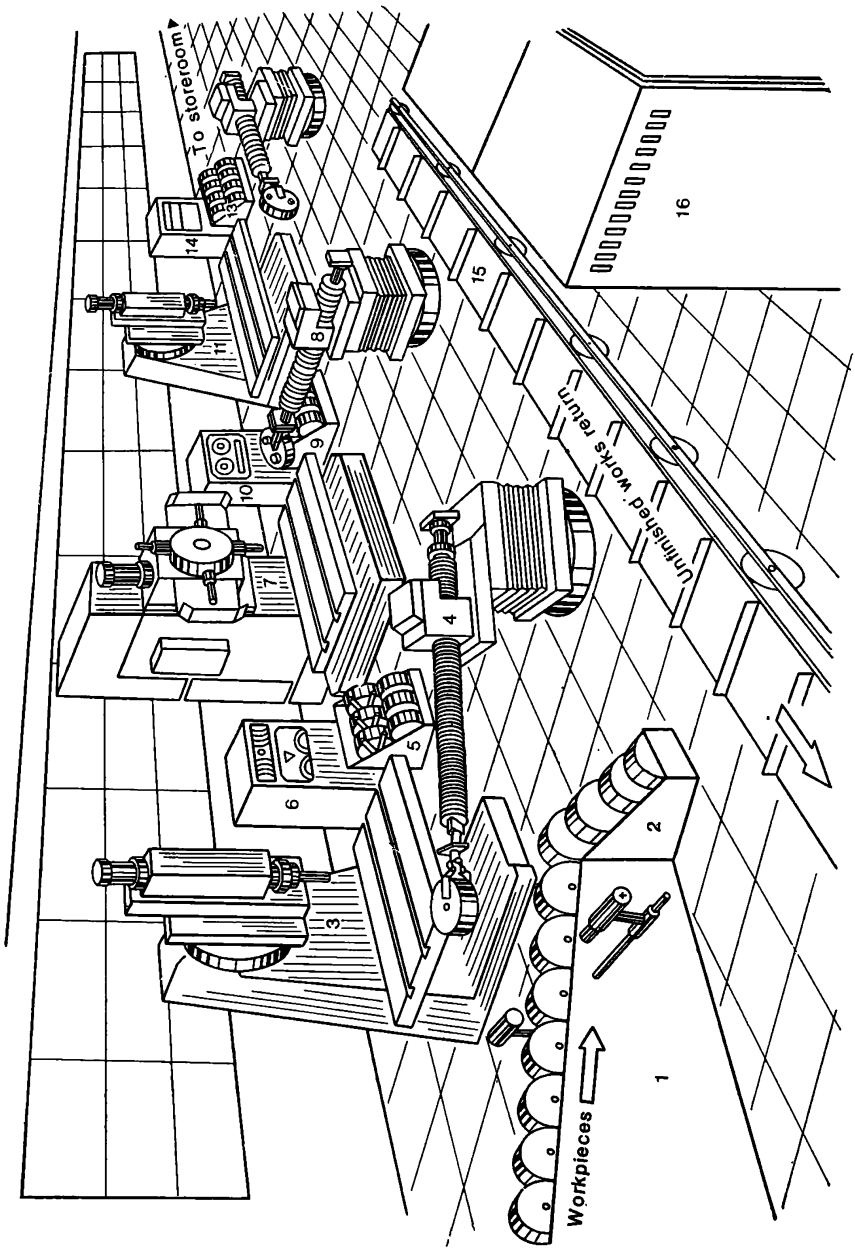


Fig. 1.1. Automatic metalworking flow line:
1 — workpiece feeder; *2* — buffer store; *3* — CNC milling tool; *4* — first manipulator; *5* — milled parts store; *6* — milling control; *7* — CNC drilling tool; *8* — second manipulator; *9* — drilled parts store; *10* — drilling control; *11* — second milling tool; *12* — third manipulator to handle finished part transport to stock room; *13* — output buffer store; *14* — second mill control; *15* — transporter for removal of wrong and recycling of unfinished parts; *16* — minicomputer to dispatch tools and manipulator controls

manipulators, that is, they are individually controlled from their own microprocessors.

On processing lines manufacturing different pieceworks, the time a robot is actuated to service the workflow may be described by a random variable distributed to a certain statistical pattern. The description assumes the times of various technological and servicing operations being statistically independent and similarly distributed. The service mechanism then may be visualized by means of a mathematical model based on queueing theory.

We shall consider mathematically the functioning of different queueing systems serviced by computer-augmented robots to devise the optimum strategy for their application.

First, we focus on choosing the set of parameters for an automatic feeder which transports workpieces to a machining tool. If the probability P of a (correctly oriented) workpiece arriving at a first machine at time τ is small then these arrivals may be treated as having a Poisson distribution. The arrival rate λ can be obtained, accordingly, from its exponential dependence

$$\exp(-\lambda\tau) = 1 - P$$

to be

$$\lambda = (1/\tau) \ln [1/(1 - P)].$$

Generally it takes a robot a random time to handle workpieces. This time is differently distributed for different operations in accordance with the arrival patterns. We shall confine ourselves with the most complicated case when the time it takes a robot to serve an arriving unit is distributed exponentially with a statistical parameter μ_0. Designating $\rho_0 = \lambda_0/\mu_0$ we may write the probability of a machine tool being idle as

$$P_0 = 1 \Big/ \sum_{n=0}^{N_0} \rho_0^n = \begin{cases} \dfrac{1-\rho_0}{1-\rho_0^{N_0+1}} & \text{at } \rho_0 \neq 1 \\ \dfrac{1}{N+1} & \text{at } \rho_0 = 1, \end{cases} \tag{1.1}$$

where N_0 is the capacity of a buffer store (palett) feeding the robotic manipulator. The probability of the store being full up is

$$P_N^0 = \rho^{N_0} P_0. \tag{1.2}$$

Accordingly, the arrival rate for the next manipulator becomes

$$\lambda_1 = \lambda_0 (1 - P_{N_0}), \tag{1.3}$$

and the rate of loss due to unserved workpieces by the first manipulator is

$$\lambda^* = \lambda P_N.$$

If we know the arrival rate to the next buffer container, then its dynamics may be evaluated by means of the expressions (1.1) and (1.2) substituting respectively λ_1, μ_1, and N_1 for λ_0, μ_0, and N_0. When a flow line involves R consequently operating manipulators, the output rate of finished items can be found as

$$\lambda_R = \lambda_0 \prod_{i=1}^{R} (1 - P_N^i), \tag{1.4}$$

where P_N^i is the probability of loss at the ith manipulator. The average count of finished items produced per unit time will be

$$\bar{K}_f = 1/\lambda_R = 1/\lambda_0 \prod_{i=1}^{R} (1 - P_N^i) \tag{1.5}$$

and the mean number of unfinished workpieces

$$\bar{K}_{uf} = 1/\lambda_0 - 1/\lambda_0 \prod_{i=1}^{R} (1 - P_N^i). \tag{1.6}$$

The latter items should reenter the workflow having been checked for unaccomplished operations. In more detail this model will be discussed below.

Striving for the optimal parameters of the flow line we devise a cost criterion subject to the constraint on the total cost of all buffer stores:

$$L = \max_{\{N_i\}} (\bar{K}_f C_f - \bar{K}_{uf} C_{uf}), \quad \sum_{i=1}^{R} N_i = Q, \tag{1.7}$$

where C_f is the cost of a finished product, C_{uf} is the cost lost due to unfinished workpieces, and Q is the total capacity of buffer stores.

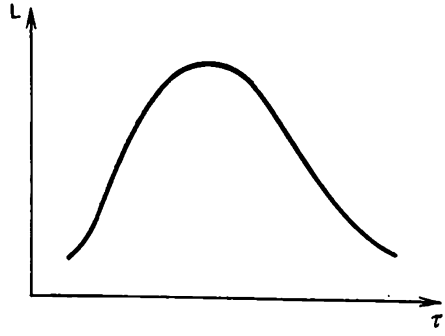


Fig. 1.2 Profit vs. mean arrival rate of workpieces at flow line

Another important parameter influencing the production rate of a flow line is the speed with which it advances. Figure 1.2 shows the profit versus the mean arrival rate of workpieces at the flow line. It

is readily seen that when the speed is high the losses due to unfinished workpieces excessively increase; with more long τ, on the other hand, the losses decrease, yet the average production rate also decreases. Therefore, τ should be included into the vector of optimized variables for the designed flow line.

Now we discuss the flow line in which the loss of workpieces is prohibitive, so that when a buffer store is overflown the line has to be stopped. If a store feeding the ith manipulator is full up, the preceding $(i-1)$st manipulator has to be stopped. The equivalent service rate of the $(i-1)$st manipulator will be

$$\bar{\mu}_{i-1} = \mu_{i-1}(1 - P_N^{(i)}) + 0\, P_N^{(i)} = \mu_{i-1}(1 - P_N^{(i)}). \tag{1.8}$$

In turn, from the expression (1.2) it follows that the probability of the ith store to overflow may be deduced from (1.1) and (1.3), i.e. from the servicing rate of the preceding manipulator. Hence in this case a flow line cannot be analysed without due allowance for the mutual interaction of the manipulators, and the robotic system should be treated as a whole by a set of equations for all states of the flow line.

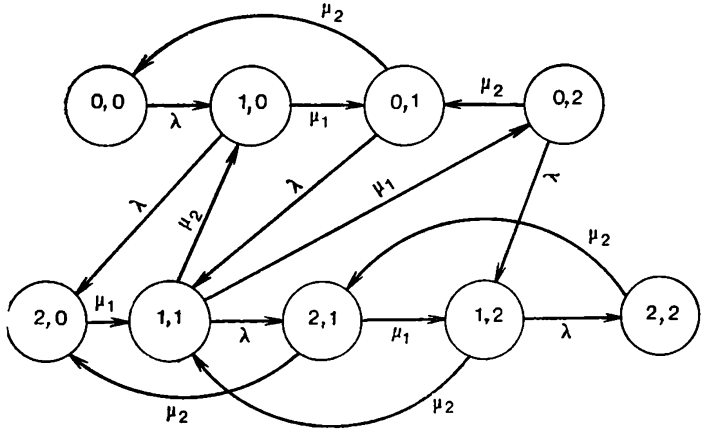


Fig. 1.3 Digraph of flow line served by master computer controlled robotic manipulators

To illustrate, Fig. 1.3 presents the graph of a flow line served by two manipulators each feeded by a unit store. The first numeral at the vertices corresponds to the number of workpieces served by the first manipulator, and the second numeral to that served by the second manipulator. A workpiece is assumed to belong to the workspace of the first manipulator if it is being served by this robot or stored in its buffer store. The state (1.1), for example, implies that the first and second robots are handling their parts, and the respective

stores are empty. The transfer of the part served by the first robot to the buffer store of the second shifts the process to state (0,2). Any new workpiece incoming into the first robot workspace shifts state (1,1) to (2,1) whereas a service finished by the second robot brings (1,1) to (1,0). Any other transients are forbidden. With reference to Fig. 1.3 we write the mean number of items produced per unit time by such a line in the form

$$\bar{K}_t = 1/\mu_2 \,(P_{0,1} + P_{0,2} + P_{1,1} + P_{1,2} + P_{2,2}), \tag{1.9}$$

where $P_{i,j}$ is the probability to find the system in a state (i, j).

In practical tasks the number of robotic manipulators R is 5 to 10 and the capacity of a buffer store N is between 10 and 50 so that the number of vertices in the graph

$$S = \prod_{i=1}^{R} (N_i + 1) \tag{1.10}$$

becomes enormously large. To avoid the difficulty we choose an approximate approach based on equivalent servicing rates for each manipulator. The probabilities $P_{N_0}^{(0)}$ and $P_{N_1}^{(1)}$ are consequently found with the expressions

$$\begin{aligned} P_{N_0}^{(0)} &= (\lambda/\mu_1 P_{N_1}^{(1)})^{N_0} \frac{1 - \lambda/\mu_1 P_{N_1}^{(1)}}{1 - (\lambda/\mu_1 P_{N_1}^{(1)})^{N_0+1}} \\ P_{N_1}^{(1)} &= \left[\frac{\lambda\,(1 - P_{N_0}^{(0)})}{\mu_2}\right]^{N_1} \frac{1 - (\lambda/\mu_2)\,(1 - P_{N_0}^{(0)})}{1 - [(\lambda/\mu_2)\,(1 - P_{N_0}^{(0)})]^{N_1+1}}. \end{aligned} \tag{1.11}$$

The resultant average production rate is

$$\bar{K}_t = 1/\lambda_0\,(1 - P_{N_0}^{(0)})\,(1 - P_{N_1}^{(1)}). \tag{1.12}$$

Note, that the so obtained expressions (1.11) and (1.12) produce but the qualitative estimates for the critical conditions of the flow line operation.

Now we shall discuss the flow line in which each machining operation is followed by testing and transporting the workpiece to the manipulator assigned in accordance with the machining quality found.

This production line may be visualized as a queueing system with feedback. Denoting θ_{ij} the probability that the workpiece is routed after the ith operation to the jth operation yields a transition matrix of flow line

$$\mathbf{T} = \begin{vmatrix} \theta_{00} & \theta_{01} & \theta_{0R} \\ \theta_{10} & \theta_{11} & \theta_{1R} \\ \cdot & \cdot & \\ \theta_{R0} & \theta_{R1} & \theta_{RR} \end{vmatrix}. \tag{1.13}$$

Using this matrix let us find the steady-state arrival rates of workpieces in the system when it is lossless and the input rate equals the throughput. Let λ_j be the total arrival rate to the jth manipulator, then

$$\lambda_j = \sum_{i=0}^{R} \lambda_i \theta_{ij}, \quad j = 0, 1, \ldots, R,$$

or in the vector form

$$\lambda_j = \mathbf{T}^T \lambda_0, \tag{1.14}$$

where λ_0 is the total source arrival rate.

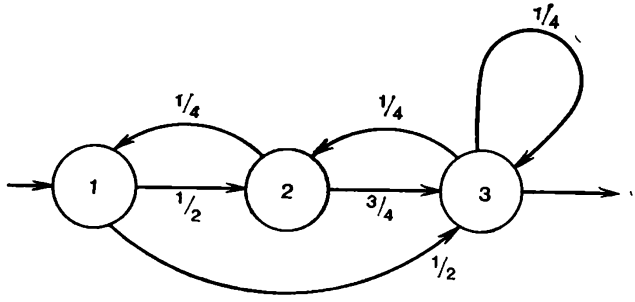


Fig. 1.4 Digraph of flow line served by three robots

To illustrate the model, consider a flow line served by three manipulators. The respective graph is presented in Fig. 1.4. Specifying the matrix T in the form

$$\mathbf{T} = \begin{vmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1/2 & 1/2 \\ 0 & 1/4 & 0 & 3/4 \\ 1/2 & 0 & 1/4 & 1/4 \end{vmatrix},$$

(1.14) expands into the following set of equations

$$\begin{aligned} -\lambda_0 + \lambda_3/2 &= 0 \\ \lambda_0 - \lambda_1 + \lambda_3/4 &= 0 \\ \lambda_1/2 - \lambda_2 + \lambda_3/4 &= 0 \\ \lambda_1/2 + 3\lambda_2/4 - 3\lambda_3/4 &= 0 \end{aligned} \tag{1.15}$$

to give upon solving

$$\lambda_1 = 9\lambda_0/7, \quad \lambda_2 = 8\lambda_0/7, \quad \lambda_3 = 2\lambda_0.$$

We have solved this problem for steady-state conditions with utilization factors for each manipulator satisfying the condition

$$\lambda_i/\mu_i < 1$$

or

$$\lambda_0 \alpha_i/\mu_i < 1.$$

This yields the constraint for the input arrival rate

$$\lambda_0 < \min_i \{\mu_i/\alpha_i\}.$$

From (1.16) we can deduce the throughput of the line. Assuming $\mu_1 = \mu_2 = \mu_3 = 1$ min^{-1} gives

$$\lambda_0 = \min \{7/9,\ 7/8,\ 1/2\} = 1/2 \text{ min}^{-1},$$

which corresponds to a mean time between arrivals of workpieces at the flow line of 2 min.

Consider now the arrival pattern of finished items from the output buffer store to a stock room. We shall suppose that they arrive packed M pieces a pack (to determine the input of containers to the stock room). The container loading can be visualized by a digraph of Fig. 1.5.

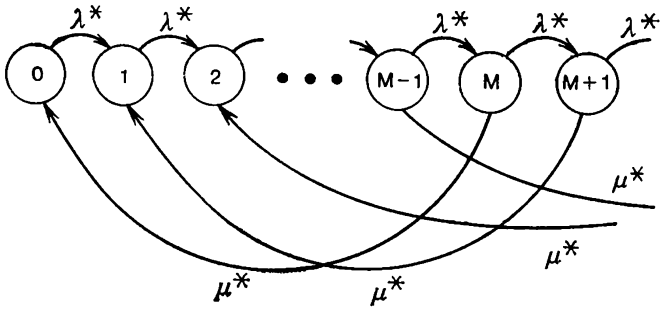


Fig. 1.5 Digraph of container loading to stock room transport

We shall derive a mathematical model of container arrivals into the stock room for a given utilization of the system when the probabilities of the first M states in the graph of Fig. 1.5 are small. To simplify our derivation mathematically we shall assume the input of M-pieces packs has a Poisson distribution. If M is a random variable having a given probability disribution M_j, then for the generating function we have approximately

$$M(x) = \sum_{j=0}^{\infty} M_j x^j. \qquad (1.16)$$

Denote a mean time interval during which the stock room contains i finished items of a given type as P_i, then we have a series

$$P(x) = \sum_{i=0}^{\infty} P_i x^i = \frac{(1-\rho)(1-x)}{1-x-x[1-M(x)]}. \tag{1.17}$$

For a unit arrival rate

$$M_i = \begin{cases} 0, \text{ at } i \neq 1 \\ 1, \text{ at } i = 1 \end{cases} \tag{1.18}$$

we have

$$M(x) = x,$$

$$P(x) = (1-\rho)/(1-\rho x). \tag{1.19}$$

Expanding $P(x)$ in a power series of x yields

$$P(x) = (1-\rho) \sum_{n=0}^{\infty} \rho^n x^n. \tag{1.20}$$

Taking advantage of (1.20) we find

$$P_i = (1-\rho)\rho^i. \tag{1.21}$$

As the stock room has a limited capacity, the probability of gathering more than N items inside it should not rise above a small value P_ε. With this in view we may write

$$P_n = \sum_{i=N+1}^{\infty} P_i = (1-\rho)\rho^{N+1} \sum_{i=0}^{\infty} \rho^i = \rho^{N+1}. \tag{1.22}$$

From the condition $\rho^{N+1} \leqslant P_\varepsilon$ we can find the capacity of the stock room, i.e. the number of cells N that satisfies the above conditions with P_ε as an upper limit

$$N = \frac{\ln P_\varepsilon}{\ln \rho} - 1. \tag{1.23}$$

Assume that the arriving lots have a Poisson distribution, that is

$$M_n = \frac{a^n}{n!} e^{-a} \tag{1.24}$$

or

$$M(x) = \exp(ax) \exp(-a), \tag{1.25}$$

then

$$P(x) = \frac{1-\rho}{1-\rho x \dfrac{1-\exp[(x-1)a]}{1-x}}. \tag{1.26}$$

For small ρ, (1.26) may be approximately written

$$P(x) = (1-\rho)\left\{1+\rho x\,\frac{1-\exp[(x-1)a]}{1-x}\right\}. \tag{1.27}$$

The expression (1.27) defines the requirements placed on the manipulator packaging the items on their final leg to a stock room.

The discussed procedure is a simple analytical tool designed to determine the rate of flow line and the required capacity of buffer stores feeding the manipulators which may operate on the individual control or master-computer (group) control basis. The obtained dependencies may be realized as the microprocessor instructions to make the operation of a flow line more versatile with respect to the type of workpieces processed, robotic manipulators used, and capacities of stock rooms available. This helps in arranging the flow line and in evaluating its performance in various working conditions.

The microprocessors used may diagnose the operation of the flow line after a sequence of operations performed within the workspace of manipulators. Should any deviations from the preset characteristics be evaluated (machining tolerances, feed store loading rate, processing time, etc.) the suggested relationships may be referenced to to automatically realign the line and to correct the functions of the manipulators.

List of Symbols

μ_0	mean service rate
λ_0	mean arrival rate
ρ_0	utilization factor
N_0	input store capacity
$P_N^{(0)}$	probability of loss at the input store
$P_{N_0}^{(0)}$	service refusal probability with N_0-capacity store
$P_{N_1}^{(1)}$	service refusal probability with N_1-capacity store

2 Mathematical Models of Robot Dynamics

V. S. Medvedev, D. Sc. (Tech.)

2.1 GENERAL

The robot arm is, as a rule, a sequence of links joined with each other and connected with other objects and drives. In many versions it may be generalized by an open kinematic loop in which the speeds and accelerations of each link can be obtained through recursive formulas by the velocity and acceleration of the preceding link and the parameters of relative travel of the links.

Robot control requires the position, speed, and acceleration of arm points. The designer of robot drives requires, in turn, the knowledge of robot kinematic behavior. To this end one may use the expressions describing the major kinematic relationships for manipulator limbs and must have at hand a set of differential equations describing the dynamics of robot.

This chapter is devoted to mathematical descriptions of robots kinematics and dynamics.

2.2 COORDINATE SYSTEMS

In the general case, we may choose the arbitrary systems of coordinates tied to the robot arm links. However, the number of parameters on which depend the matrices of transformation from one frame to another should be kept to a minimum. Consider at first a method to represent the link coordinate systems, which proved suitable for a number of problems in manipulator design [42]. In what follows we shall discuss only those robots whose arm linkage consists of revolute and prismatic (sliding) pairs of connectivity one, i.e. having one degree of freedom. For both types of kinematic pairs we introduce the notion of the axis of the pair. In a revolute pair, this will be the axis of relative link rotation; in a prismatic pair, a direct line parallel to the direction of relative sliding motions.

We enumerate all the links of the linkage one by one starting with the link connected to the frame to which we assign the number 1 up to the terminal link, i.e. hand, or end effector, to which we assign the number N. The base will be assumed unmovable in the inertial reference frame until specified otherwise.

We shall direct the Z axis of each link coordinate system along the axis of the kinematic pair connecting this link to the next in the

linkage (Fig. 2.1). The terminal link (end effector) has no following link, therefore the axis Z_N can be directed arbitrarily, usually along the axis of the hand orientation (Fig. 2.2). The axes X_i ($i = 1, 2, \ldots, N$) are chosen so that they be perpendicular to both Z_{i-1} and Z_i. The axes Y_i form with Z_i and X_i the right-hand orthogonal coordinate systems.

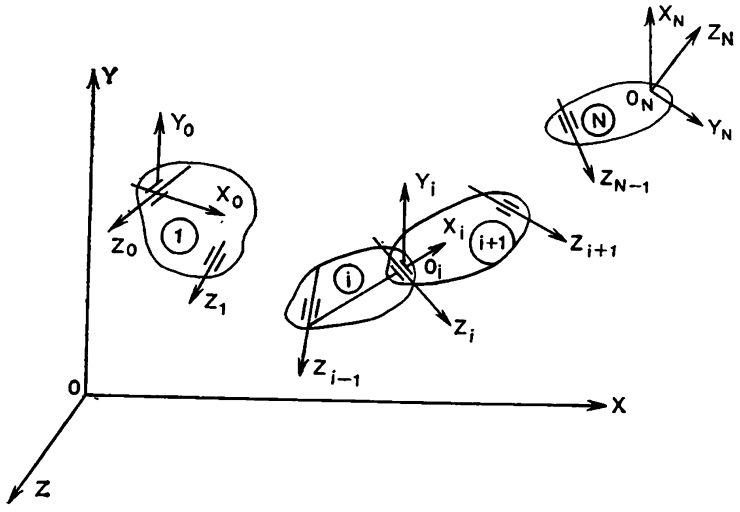


Fig. 2.1 Coordinate systems

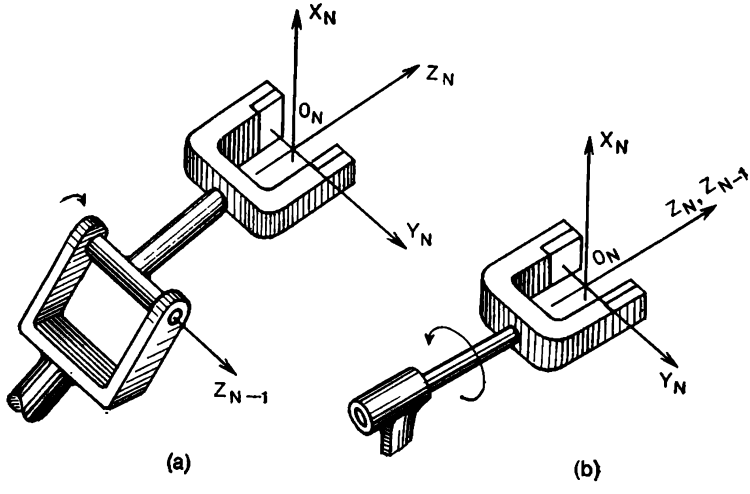


Fig. 2.2 Hand-tied coordinate syste

Now, the orientation of the base-tied coordinate system $(0_0X_0Y_0Z_0)$ is left to define. If motion of the mechanism will be discussed in this very system, then the axis X_0 can be directed arbitrarily. If another, more convenient system exists, say $0XYZ$, then it can be chosen as a reference frame, but X_0 should then be oriented so as to be perpendicular to both Z and Z_0.

Choosing the coordinate system in the above manner we may pinpoint four parameters completely defining the relative position of any two sequential coordinate systems and, hence, the position of the respective links. Two of these parameters are constants decided by the chosen arm structure, and one of the two left is a variable and

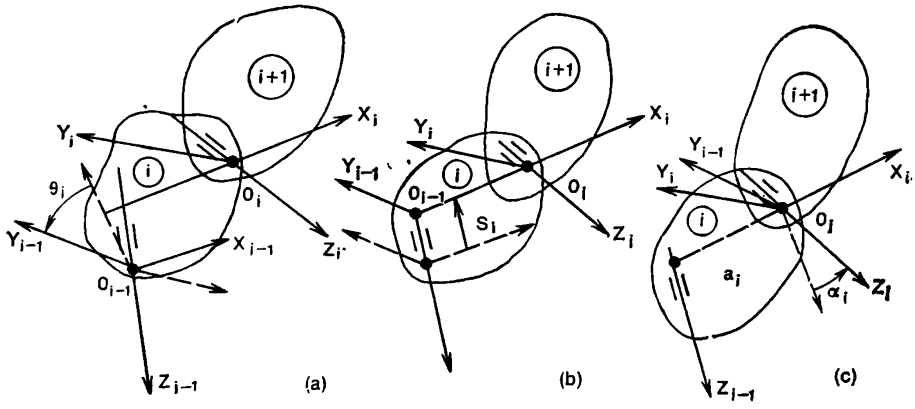


Fig. 2.3 Illustrating transition from $(i-1)$st to ith frame of reference

may be chosen for a generalized coordinate according as the pair is revolute or prismatic. To prove, we may always change from the $(i-1)$st to the ith system by a rotation, two-fold translation and another rotation performed in the following order (Fig. 2.3):

(i) rotation about the axis Z_{i-1} through an angle θ_i into the position $0_{i-1}X'Y'Z_{i-1}$ where the axis X' is parallel to the axis Y_i;

(ii) sliding along the axis Z_{i-1} for a length S_i into the position $0'XYZ$ until the axes X' and X_i become co-linear;

(iii) sliding along the axis X_i for a length a_i until the origin $0''$ coincides with the origin of the system $(0_iX_iY_iZ_i)$ in a new position $(0_iX_iY''Z'')$;

(iv) rotation through an angle α_i about the axis X_i to coincide with the ith coordinate system.

If the coordinates of a vector r in the system $0_{i-1}X_{i-1}Y_{i-1}Z_{i-1}$ are x_{i-1}, y_{i-1}, and z_{i-1}, then the above transformation steps yield

sequentially

(i) $$\begin{vmatrix} x' \\ y' \\ z' \end{vmatrix} = \begin{vmatrix} \cos\theta_i & \sin\theta_i & 0 \\ -\sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} x_{i-1} \\ y_{i-1} \\ z_{i-1} \end{vmatrix},$$

(ii) $$\begin{vmatrix} x \\ y \\ z \end{vmatrix} = \begin{vmatrix} \cos\theta_i & \sin\theta_i & 0 \\ -\sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} x_{i-1} \\ y_{i-1} \\ z_{i-1} \end{vmatrix} - \begin{vmatrix} 0 \\ 0 \\ S_i \end{vmatrix},$$

(iii) $$\begin{vmatrix} x'' \\ y'' \\ z'' \end{vmatrix} = \begin{vmatrix} \cos\theta_i & \sin\theta_i & 0 \\ -\sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} x_{i-1} \\ y_{i-1} \\ z_{i-1} \end{vmatrix} - \begin{vmatrix} 0 \\ 0 \\ S_i \end{vmatrix} - \begin{vmatrix} a_i \\ 0 \\ 0 \end{vmatrix},$$

(iv) $$\begin{vmatrix} x_i \\ y_i \\ z_i \end{vmatrix} = \begin{vmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha_i & \sin\alpha_i \\ 0 & -\sin\alpha_i & \cos\alpha_i \end{vmatrix}$$

$$\times \left\{ \begin{vmatrix} \cos\theta_i & \sin\theta_i & 0 \\ -\sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} x_{i-1} \\ y_{i-1} \\ x_{i-1} \end{vmatrix} - \begin{vmatrix} 0 \\ 0 \\ S_i \end{vmatrix} - \begin{vmatrix} a_i \\ 0 \\ 0 \end{vmatrix} \right\}$$

Thus, the coordinates $[x_i, y_i, z_i]$ of vector $\mathbf{r}$ in the system $0_iX_iY_iZ_i$ are

$$\begin{aligned} x_i &= x_{i-1}\cos\theta_i + y_{i-1}\sin\theta_i - a_i \\ y_i &= x_{i-1}\sin\theta_i\cos\alpha_i + y_{i-1}\cos\theta_i\cos\alpha_i + z_{i-1}\sin\alpha_1 - S_i\sin\alpha_i \\ z_i &= x_{i-1}\sin\theta_i\sin\alpha_i - y_{i-1}\cos\theta_i\sin\alpha_i + S_i\cos\alpha_i \end{aligned} \qquad (2.1)$$

where $[x_{i-1}, y_{i-1}, z_{i-1}]$ are the coordinates of the vector $\mathbf{r}$ in the $(i-1)$st system.

The reverse transformation is

$$\begin{aligned} x_{i-1} &= x_i\cos\theta_i - y_i\sin\theta_i\cos\alpha_i + z_i\sin\theta_i\sin\alpha_i + a_i\cos\theta_i \\ y_{i-1} &= x_i\sin\theta_i + y_i\cos\theta_i\cos\alpha_i - z_i\cos\theta_i\sin\alpha_i + a_i\sin\theta_i \\ z_{i-1} &= y_i\sin\alpha_i + z_i\cos\alpha_i + S_i \end{aligned} \qquad (2.2)$$

In the following we adopt a convention to supply a vector with superscript, say i, if it is defined in the ith system of coordinates. Thus, $\mathbf{r}_j^i$ is a vector $\mathbf{r}_j$ specified in the ith system of coordinates, and $\mathbf{r}_j^0$ is the same vector defined by its coordinates in the inertial system $(0_0X_0Y_0Z_0)$.

Employing this convention we may write the above in a more compact form

$$\mathbf{r}^i = \tau_{i,i-1}\mathbf{r}^{i-1} - \mathbf{l}_i^i, \qquad (2.3)$$

where

$$\tau_{i,\,i-1} = \begin{vmatrix} \cos\theta_i & \sin\theta_i & 0 \\ -\sin\theta_i\cos\alpha_i & \cos\theta_i\cos\alpha_i & \sin\alpha_i \\ \sin\theta_i\sin\alpha_i & -\cos\theta_i\sin\alpha_i & \cos\alpha_i \end{vmatrix} \tag{2.4}$$

is the rotation matrix transforming the $(i-1)$st to the ith system of coordinates;

$$\mathbf{l}_i^i = \begin{vmatrix} a_i \\ S_i\sin\alpha_i \\ S_i\cos\alpha_i \end{vmatrix} \tag{2.5}$$

is the vector from the origin of the $(i-1)$st to the origin of the ith system of coordinates. It is specified in the ith system and defines a translational shift in changing from the ith to $(i-1)$st system of coordinates.

Accordingly, changing back from the ith system to the $(i-1)$st one we get

$$\mathbf{r}^{i-1} = \tau_{i-1,i}\mathbf{r}^i + \mathbf{l}_i^{i-1}, \tag{2.6}$$

where

$$\tau_{i-1,\,i} = \begin{vmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i \\ 0 & \sin\alpha_i & \cos\alpha_i \end{vmatrix}, \tag{2.7}$$

$$\mathbf{l}_i^{i-1} = \begin{vmatrix} a_i\cos\theta_i \\ a_i\sin\theta_i \\ S_i \end{vmatrix}. \tag{2.8}$$

Note, that

$$\tau_{i-1,\,i} = \tau_{i,i-1}^{-1} = \tau_{i,i-1}^{\mathrm{T}} \tag{2.9}$$

because this matrix defines a transformation from one orthogonal basis to the other, being orthogonal in itself. From (2.2) and (2.6) it follows that

$$\mathbf{r}^{i-1} = \tau_{i-1,i}(\mathbf{r}^i + \mathbf{l}_i^i). \tag{2.10}$$

Of four parameters θ_i, s_i, a_i, and α_i involved in (2.4) and (2.5) [or (2.7) and (2.8)], two α_i and a_i are always constant and decided by the design of links. One of the two parameters, θ_i or S_i, is variable. For the revolute joint, the quantity θ_i defines the angle of relative rotation of the links $i-1$ and i, while the linear sliding S_i is constant. For the sliding joint, on the contrary, the angle θ_i is constant, while S_i varies.

We shall include the sole variable of the ith movable joint, θ_i or S_i, into the set of the generalized coordinates of the mechanism, labelling it q_i ($i = 1, 2, \ldots, N$).

For the screw joint, the dependence of θ_i and S_i on the generalized coordinates is more complex. Write for the general case

$$\theta_i = n_\theta q_i,$$
$$S_i = n_s q_i.$$

To illustrate, for $\theta_i = q_i$, $n_\theta = 1$ and $n_s = h/2\pi$; if we take linear translation for the generalized coordinate q_i, then $n_s = 1$ and $n_\theta = 2\pi/h$, where h is the pitch.

By consecutively changing from the ith to the $(i-1)$st, $(i-2)$nd and, finally, to the inertial coordinate system we have

$$\begin{aligned}
&\mathbf{r}^{i-1} = \tau_{i-1,\,i}\,(\mathbf{r}_i^i + \mathbf{l}_i^i),\\
&\mathbf{r}^{i-2} = \tau_{i-2,\,i-1}\,[\tau_{i-1,\,i}\,(\mathbf{r}_i^i + \mathbf{l}_i^i) + \mathbf{l}_{i-1}^{i-1}],\\
&\qquad\qquad \ldots\ldots\ldots\ldots\ldots\\
&\mathbf{r}^0 = \Big(\prod_{j=1}^{i} \tau_{j-1,\,j}\Big)\,\mathbf{r}_i^i + \sum_{k=1}^{1}\Big(\prod_{j=1}^{k} \tau_{j-1,\,j}\Big)\,\mathbf{l}_k^k.
\end{aligned} \tag{2.11}$$

Denote

$$\tau_{0i}(\theta_1, \theta_2, \ldots, \theta_i) = \prod_{j=1}^{i} \tau_{i-1,\,j}(\theta_j). \tag{2.12}$$

Here, τ_{0i} is the matrix of rotation which results in relating the ith to the 0th system of coordinates by i successive transformations. With this we may write

$$\mathbf{r}^0 = \tau_{0i}\mathbf{r}^i + \sum_{j=1}^{i} \tau_{j-1,\,j}\mathbf{r}_j. \tag{2.13}$$

From the orthogonality of the transformation matrices

$$\tau_{i0} = \tau_{0i}^{\mathrm{T}}. \tag{2.14}$$

Consider now the rules of differentiating the transformation matrices $\tau_{i-1,\,i}$ and vectors $\mathbf{l}_i^i$.

For a revolute kinematic pair of connectivity one,

$$\begin{aligned}
&\theta_i = q_i = \text{variable},\\
&S_i = \text{constant}.
\end{aligned} \tag{2.15}$$

Differentiating (2.4) with respect to time, we get

$$\begin{aligned}
\dot{\tau}_{i-1,\,i}(q_i) &= \begin{vmatrix} -\sin\theta_i & -\cos\theta_i\cos\alpha_i & \cos\theta_i\sin\alpha_i \\ \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i \\ 0 & 0 & 0 \end{vmatrix} \dot{q}_i\\
&= \Theta_\theta \tau_{i-1,\,i}(q_i)\,\dot{q}_i,
\end{aligned} \tag{2.16}$$

where

$$\Theta_\theta = \begin{vmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{vmatrix}.$$

From (2.15) it follows that

$$\dot{\mathbf{i}}_i^i = 0. \tag{2.17}$$

Denoting the unit vector directed in the axis Z_{i-1} of the $(i-1)$st coordinate frame as $\mathbf{e}_{i-1}^{i-1} = [0\ 0\ 1]^T$ and bringing in correspondence with a three-dimensional vector $\mathbf{a} = {}_i[a_x a_y a_z]^T$ the 3×3 matrix

$$\lambda(\mathbf{a}) = \begin{vmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{vmatrix} \tag{2.18}$$

produces (2.16) in the form

$$\dot{\tau}_{i-1,\,i} = \lambda(\mathbf{e}_{i-1}^{i-1})\tau_{i-1,\,i}q_i. \tag{2.19}$$

Invoking the matrix $\lambda(\mathbf{a})$ we may represent the vector product $\mathbf{c} = \mathbf{a} \times \mathbf{b}$ as the matrix by column matrix product

$$\mathbf{c} = \lambda(\mathbf{a})\mathbf{b}. \tag{2.20}$$

To prove, with the reference to (2.18)

$$\lambda(\mathbf{a})\,\mathbf{b} = \begin{vmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{vmatrix} \begin{vmatrix} b_x \\ b_y \\ b_z \end{vmatrix} = \begin{vmatrix} a_y b_z - a_z b_y \\ a_z b_x - a_x b_z \\ a_x b_y - a_y b_x \end{vmatrix}, \tag{2.21}$$

thus, the elements of the resultant column matrix are equal to the vector product coordinates.

For the product (2.20), vectors $\mathbf{a}$ and $\mathbf{b}$, a scalar α, and an orthogonal matrix τ, the following relations are valid

$$\lambda(\alpha\mathbf{a}) = \alpha\lambda(\mathbf{a}), \tag{2.22}$$

$$\lambda(\mathbf{a})\,\mathbf{a} = 0, \tag{2.23}$$

$$\lambda(\mathbf{a})\mathbf{b} = -\mathbf{b}\lambda(\mathbf{a}), \tag{2.24}$$

$$\frac{d}{dt}\lambda(\mathbf{a}) = \lambda\left(\frac{d\mathbf{a}}{dt}\right), \tag{2.25}$$

$$\lambda(\tau\mathbf{a}) = \tau\lambda(\mathbf{a})\tau^T, \tag{2.26}$$

$$\tau\lambda(\mathbf{a}) = \lambda(\tau\mathbf{a})\tau, \tag{2.27}$$

$$\lambda\left(\sum_{i=1}^{n}\mathbf{a}_i\right) = \sum_{i=1}^{n}\lambda(\mathbf{a}_i), \tag{2.28}$$

$$\lambda[\lambda(\mathbf{a})\mathbf{b}] = \lambda(\mathbf{a})\lambda(\mathbf{b}) - \lambda(\mathbf{b})\lambda(\mathbf{a}). \tag{2.29}$$

In view of (2.27) the equation (2.19) can be transformed to be

$$\dot{\tau}_{i-1,\,i} = \tau_{i-1,\,i}\lambda\,(\tau_{i,\,i-1}\mathbf{e}_{i-1}^{i-1})\,\dot{q}_i$$
$$= \tau_{i-1,\,i}\lambda\,(\mathbf{e}_{i-1}^{i})\,\dot{q}_i = \tau_{i-1,\,i}\lambda\,(\mathbf{v}_i\dot{q}_i). \tag{2.30}$$

where

$$\mathbf{e}_{i-1}^{i} = \tau_{i,i-1}\mathbf{e}_{i-1}^{i-1} = \mathbf{v}_i = [0\ \sin\alpha_i\ \cos\alpha_i]^{\mathrm{T}} \tag{2.31}$$

is the unit vector in the axis Z_{i-1} defined in the ith frame.

For a prismatic pair of connectivity one

$$q_i = S_i = \text{variable},$$
$$\theta_i = \text{constant}. \tag{2.32}$$

In this case the matrix $\tau_{i-1,i}$ is constant so that

$$\dot{\tau}_{i-1,\,i} = 0. \tag{2.33}$$

Subject to (2.32) the derivative of (2.5) becomes

$$\dot{\mathbf{l}}_i^i = [0\ \sin\ \alpha_i\ \cos\alpha_i]^{\mathrm{T}}\dot{q}_i. \tag{2.34}$$

For the screw kinematic connection of connectivity one, assuming linear translation as a generalized coordinate we have

$$S_i = q_i = \text{variable}, \tag{2.35}$$
$$\theta_i = (2\pi/h)q_i.$$

From (2.30)-(2.32) and (2.34) we find

$$\dot{\tau}_{i-1,\,i} = \tau_{i-1,\,i}\lambda\left(\mathbf{v}_i\frac{2\pi}{h}\,\dot{q}_i\right), \tag{2.36}$$

$$\dot{\mathbf{l}}_i^i = \mathbf{v}_i\dot{q}_i. \tag{2.37}$$

Introducing two parameters $\sigma_{\theta i}$ and σ_{si} sensitive to the type of the ith joint and such that

$$\sigma_{\theta i} = \begin{cases} 1 \text{ for a revolute joint} \\ 0 \text{ for a sliding joint} \\ 2\pi/h \text{ for a screw joint} \end{cases} \tag{2.38}$$

$$\sigma_{si} = \begin{cases} 0 \text{ for a revolute joint} \\ 1 \text{ for a sliding or screw joint,} \end{cases}$$

the equations for the derivatives of the joints merge to be for all the above types

$$\dot{\tau}_{i-1,i} = \sigma_{\theta i}\tau_{i-1,i}\lambda\,(\mathbf{v}_i\dot{q}_i) = \tau_{i-1,\,i}\lambda\,(\sigma_{\theta i}\mathbf{v}_i\dot{q}_i), \tag{2.39}$$

$$\dot{\mathbf{l}}_i^i = \sigma_{si}\mathbf{v}_i\dot{q}_i. \tag{2.40}$$

Now we may derive a formula for the time derivative of rotation matrix τ_{0i} referring the ith coordinate system to the inertial frame. Differentiating (2.12) gives

$$\dot{\tau}_{0i} = \sum_{m=1}^{i} \Big(\prod_{j=1}^{m-1} \tau_{j-1,\,j} \Big) \dot{\tau}_{m-1,\,m} \Big(\prod_{j=m+1}^{i} \Big) \tau_{j-1,\,j}$$

$$= \sum_{m=1}^{i} \tau_{0,\,m-1} \dot{\tau}_{m-1,\,m} \tau_{mi}. \tag{2.41}$$

Recalling the equations (2.41), (2.39), (2.27) and (2.28) we obtain

$$\dot{\tau}_{0i} = \sum_{m=1}^{i} \tau_{0,\,m-1} \tau_{m-1,\,m} \lambda (\sigma_{\theta m} \nu_m \dot{q}_m) \tau_{mi}$$

$$= \sum_{m=1}^{i} \tau_{0m} \lambda (\sigma_{\theta m} \nu_m \dot{q}_m) \tau_{mi}$$

$$= \tau_{0i} \sum_{m=1}^{i} \tau_{im} \lambda (\sigma_{\theta m} \nu_m \dot{q}_m) \tau_{mi}$$

$$= \tau_{0i} \sum_{m=1}^{i} \lambda (\tau_{im} \sigma_{\theta m} \nu_m \dot{q}_m)$$

$$= \tau_{0i} \lambda \Big(\sum_{m=1}^{i} \tau_{im} \sigma_{\theta m} \nu_m \dot{q}_m \Big). \tag{2.42}$$

The expressions we have derived in this section will be used in the following to deduce kinematic and dynamic equations for the robot arm.

2.3 BASIC KINEMATIC EQUATIONS

Kinematic equations describe motion of bodies with no account of its cause, i.e. forces and moments. For robot arm motion, these equations define linear and angular velocities of arm links in both relative and absolute motion.

To obtain relative angular and linear velocity we denote as $\tilde{\omega}_i$ the vector of angular velocity of the ith link relative to the $(i-1)$st, that is, of the angular velocity of the $(0_i X_i Y_i Z_i)$ with respect to the $(0_{i-1} X_{i-1} Y_{i-1} Z_{i-1})$ frame. Recalling that this vector (for a revolute joint) is directed along the Z_{i-1} axis, we have

$$\tilde{\omega}_{i-1}^{i-1} = [0\ 0\ 1]^{\mathrm{T}} \dot{q}_i. \tag{2.43}$$

From the equalities (2.1) and (2.43) it follows that

$$\tilde{\omega}_i^i = \tau_{i,i-1} [0\ 0\ 1]^{\mathrm{T}} \dot{q}_i$$

$$= [0 \sin \alpha_i \cos \alpha_i]^{\mathrm{T}} \dot{q}_i = \nu_i \dot{q}_i, \tag{2.44}$$

where $\nu_i = [0 \sin \alpha_i \cos \alpha_i]^T$ is the directional cosine vector of the Z_{i-1} axis with respect to the ith frame. When the $(i-1)$st and ith links are connected into a prismatic pair with no rotation between them,

$$\tilde{\omega}_i^i = 0. \tag{2.45}$$

For the screw joint, if we take linear displacement as the generalized coordinate, then

$$\tilde{\omega}_i^i = (2\pi/h)\nu_i \dot{q}_i. \tag{2.46}$$

With due account of (2.38), the formulas (2.43), (2.45) and (2.46) may be reduced into a single expression

$$\tilde{\omega}_i^i = \sigma_{\theta i} \nu_i \dot{q}_i. \tag{2.47}$$

The equation (2.39) with reference to (2.5) may be rewritten

$$\dot{\tau}_{i-1,i} = \tau_{i-1,i} \lambda (\tilde{\omega}_i^i). \tag{2.48}$$

Now we shall deduce the expressions for the relative velocity $\tilde{v}_i$ of sliding motion of the ith link with respect to the $(i-1)$st. Since the frames are chosen so that the relative motion in the ith joint is parallel to the axis of the joint, we may write for prismatic and screw kinematic pairs

$$\tilde{v}_i^i = \nu_i \dot{q}_i. \tag{2.49}$$

Note that here ν_i is the unit vector directed along the axis Z_{i-1} and specified in the ith frame, and $q_i = S_i$.

In the revolute kinematic pair, translational displacement is absent, therefore

$$\tilde{v}_i^i = 0. \tag{2.50}$$

Recalling (2.38), we join (2.49) and (2.50) into the single formula

$$\tilde{v}_i^i = \sigma_{si} \nu_i \dot{q}_i. \tag{2.51}$$

Now (2.40) may be rewritten as

$$\dot{l}_i^i = \tilde{v}_i^i. \tag{2.52}$$

The absolute angular velocity of the ith link is found as a sum of relative angular velocities of the links 1 through i. Thus, with reference to the inertial frame and with substitution from (2.47) we get

$$\omega_i^0 = \sum_{j=1}^{i} \tilde{\omega}_j^0 = \sum_{j=1}^{i} \sigma_{\theta j} \tau_{0j} \nu_j \dot{q}_j \tag{2.53}$$

or with respect to the ith link frame

$$\omega_i^i = \sum_{j=1}^{i} \tau_{ij}\sigma_{\theta j}\nu_j \dot{q}_j. \tag{2.54}$$

Subject to (2.53) the equation (2.42) may be written as

$$\dot{\tau}_{0i} = \tau_{0i}\lambda(\omega_i^i). \tag{2.55}$$

Let us now find the absolute velocity of a reference point fixed in the ith link frame and defined by the position vector $\mathbf{r}_i^i$ = constant. The position vector with respect to the base-tied frame of reference will be

$$\mathbf{r}_i^0 = \tau_{0i}\mathbf{r}_i^i + \sum_{j=1}^{i} \tau_{ij}\mathbf{l}_j^j. \tag{2.56}$$

Taking a derivative with respect to time we get

$$\mathbf{v}_i^0 = \dot{\mathbf{r}}_i^0 = \dot{\tau}_{0i}\mathbf{r}_i^i + \sum_{j=1}^{i} \dot{\tau}_{0j}\mathbf{l}_j^j + \sum_{j=1}^{i} \tau_{0j}\dot{\mathbf{l}}_j^j, \tag{2.57}$$

where $\mathbf{v}_i^0$ is the absolute velocity of the point specified by the position vector $\mathbf{r}_i^i$. Making use of the expressions (2.10) and (2.13) gives

$$\begin{aligned}\mathbf{v}_i^0 &= \tau_{0i}\lambda(\omega_i^i)\,\mathbf{r}_i^i + \sum_{j=1}^{i} \tau_{0j}\lambda(\omega_j^j)\,\mathbf{l}_j^j + \sum_{j=1}^{i} \tau_{0j}\tilde{\mathbf{v}}_j^j \\ &= \tau_{0i}\lambda\Big(\sum_{j=1}^{i} \tau_{ij}\sigma_{\theta j}\nu_j\dot{q}_j\Big)\,\mathbf{r}_i^i \\ &\quad + \sum_{j=1}^{i} \tau_{0i}\lambda\Big(\sum_{m=1}^{j} \tau_{jm}\sigma_{\theta m}\dot{q}_m\Big)\,\mathbf{l}_j^j + \sum_{j=1}^{i} \tau_{0j}\sigma_{sj}\nu_j\dot{q}_j.\end{aligned} \tag{2.58}$$

Premultiplying (2.58) by the matrix $\tau_{i0} = \tau_{0i}^T$ yields the absolute velocity of linear motion referred to frame i

$$\mathbf{v}_i^i = \lambda(\omega_i^i)\,\mathbf{r}_i^i + \sum_{j=1}^{i} \tau_{ij}\lambda(\omega_j^j)\,\mathbf{l}_j^j + \sum_{j=1}^{i} \tau_{ij}\sigma_{sj}\nu_j\dot{q}_j. \tag{2.59}$$

Using (2.24) the equations (2.58) and (2.59) may be written as

$$\begin{aligned}\mathbf{v}_i^i &= \tau_{0i}\Big[-\lambda(\mathbf{r}_i^i)\sum_{j=1}^{i} \tau_{ij}\sigma_{\theta j}\nu_j\dot{q}_j \\ &\quad - \sum_{j=1}^{i} \tau_{ij}\lambda(\mathbf{l}_j^j)\sum_{m=1}^{j} \tau_{im}\sigma_{\theta m}\nu_m\dot{q}_m + \sum_{j=1}^{i} \tau_{ij}\sigma_{sj}\nu_j\dot{q}_j\Big],\end{aligned} \tag{2.60}$$

$$\begin{aligned}\mathbf{v}_i^i &= -\lambda(\mathbf{r}_i^i)\sum_{j=1}^{i} \tau_{ij}\sigma_{\theta j}\nu_j\dot{q}_j - \sum_{j=1}^{i} \tau_{ij}\lambda(\mathbf{l}_j^j) \\ &\quad \times \sum_{m=1}^{j} \tau_{im}\sigma_{\theta m}\nu_m\dot{q}_m + \sum_{j=1}^{i} \tau_{ij}\sigma_{sj}\nu_j\dot{q}_j.\end{aligned} \tag{2.61}$$

To compute the inertial acceleration of the ith link we differentiate (2.53) with respect to time, and substituting from (2.55), obtain

$$\begin{aligned}\varepsilon_i^0 = \dot{\omega}_i^0 &= \sum_{j=1}^{i} \sigma_{\theta j}\dot{\tau}_{0j}\nu_j\dot{q}_j + \sum_{j=1}^{i} \sigma_{\theta j}\tau_{0j}\nu_j\ddot{q}_j \\ &= \sum_{j=1}^{i} \tau_{0j}\lambda(\omega_j^j)\,\sigma_{\theta j}\nu_j\dot{q}_j + \sum_{j=1}^{i} \sigma_{\theta j}\tau_{0j}\nu_j\ddot{q}_j \\ &= \sum_{j=1}^{i} \tau_{0j}\lambda(\omega_j^j)\,\tilde{\omega}_j^j + \sum_{j=1}^{i} \tau_{0j}\tilde{\varepsilon}_j^j, \end{aligned} \tag{2.62}$$

where

$$\tilde{\varepsilon}_j^j = \sigma_{\theta j}\nu_j\ddot{q}_j \tag{2.63}$$

is the relative angular acceleration of the ith link rotating with respect to the $(i-1)$st.

Premultiplying (2.63) by the matrix $\tau_{ij} = \tau_{0i}^{\mathrm{T}}$ yields the absolute angular acceleration of the ith link with respect to the ith system of coordinates

$$\begin{aligned}\varepsilon_i^i &= \sum_{j=1}^{i} \tau_{ij}\lambda(\omega_j^j)\,\sigma_{\theta j}\nu_j\dot{q}_j + \sum_{j=1}^{i} \tau_{ij}\sigma_{\theta j}\nu_j\ddot{q}_j \\ &= \sum_{j=1}^{i} \tau_{ij}\lambda(\omega_j^j)\,\tilde{\omega}_j^j + \sum_{j=1}^{i} \tau_{ij}\tilde{\varepsilon}_j^j. \end{aligned} \tag{2.64}$$

Recalling that from (2.54)

$$\omega_j^j = \sum_{m=1}^{j} \tau_{jm}\sigma_{\theta m}\nu_m\dot{q}_m$$

and taking account of (2.28) and (2.22) we have

$$\lambda(\omega_j^j) = \sum_{m=1}^{j} \lambda(\tau_{jm}\nu_m)\,\sigma_{\theta m}\dot{q}_m. \tag{2.65}$$

Substituting this equation into (2.62) and (2.65) we get, respectively

$$\varepsilon_i^0 = \sum_{j=1}^{i} \tau_{0j} \sum_{m=1}^{j} \lambda(\tau_{jm}\nu_m)\,\nu_j\sigma_{\theta j}\sigma_{\theta m}\dot{q}_j\dot{q}_m + \sum_{i=1}^{i} \tau_{0j}\sigma_{\theta j}\nu_j\ddot{q}_j, \tag{2.66}$$

$$\varepsilon_i^i = \sum_{j=1}^{i} \tau_{ij} \sum_{m=1}^{j} \lambda(\tau_{jm}\nu_m)\,\nu_j\sigma_{\theta j}\sigma_{\theta m}\dot{q}_m\dot{q}_j + \sum_{j=1}^{i} \tau_{ij}\sigma_{\theta j}\nu_j\ddot{q}_j. \tag{2.67}$$

It is seen that the vector of absolute angular acceleration is a function of second derivatives of the generalized coordinates $\ddot{q}_j$ and of products of the first derivatives $\dot{q}_j\dot{q}_m$.

Now we will find the absolute linear acceleration $\mathbf{w}_i^0$ of an arbitrary point on the ith link, defined by the position vector $\mathbf{r}_i^i$, referred to the inertial frame. To this end, perform differentiation of (2.60) with respect to time to get

$$\begin{aligned}
\mathbf{w}_i^0 = & -\dot{\tau}_{0i}\lambda(\mathbf{r}_i^i)\sum_{j=1}^{i}\tau_{ij}\sigma_{\theta j}\nu_j\dot{q}_j - \lambda(\tau_{0i}\mathbf{r}_i^i)\sum_{j=1}^{i}\dot{\tau}_{0j}\sigma_{\theta j}\nu_j\dot{q}_j \\
& - \sum_{j=1}^{i}\dot{\tau}_{0j}\lambda(\mathbf{l}_j^j)\sum_{m=1}^{j}\tau_{jm}\sigma_{\theta m}\nu_m\dot{q}_m - \tau_{0i}\lambda(\mathbf{r}_i^i)\sum_{j=1}^{i}\tau_{ij}\sigma_{\theta j}\nu_j\ddot{q}_j \\
& - \sum_{j=1}^{i}\tau_{0j}\lambda(\dot{\mathbf{l}}_j^j)\sum_{m=1}^{j}\tau_{jm}\sigma_{\theta m}\nu_m\dot{q}_m - \sum_{j=1}^{i}\lambda(\tau_{0j}\mathbf{l}_j^j)\sum_{m=1}^{j}\dot{\tau}_{0m}\sigma_{\theta m}\nu_m\dot{q}_m \\
& - \sum_{j=1}^{i}\tau_{0j}\mathbf{l}_j^j\sum_{m=1}^{j}\tau_{jm}\sigma_{\theta m}\nu_m\ddot{q}_m + \sum_{j=1}^{i}\dot{\tau}_{0j}\sigma_{sj}\nu_j\dot{q}_j \\
& + \tau_{0i}\sum_{j=1}^{i}\tau_{ij}\sigma_{sj}\nu_j\ddot{q}_j. \qquad (2.68)
\end{aligned}$$

In the differentiation we have taken account of (2.26) whereby

$$\tau_{0i}\lambda(\mathbf{r}_i^i)\tau_{ij} = \lambda(\tau_{0i}\mathbf{r}_i^i)\tau_{0i}\tau_{ij},$$

$$\tau_{0j}\lambda(\mathbf{l}_j^j)\tau_{jm} = \lambda(\tau_{0j}\mathbf{l}_j^j)\tau_{0j}\tau_{jm}.$$

Making use of (2.54), (2.55), (2.40) and (2.26) compute the terms of (2.68)

$$\begin{aligned}
& -\dot{\tau}_{0i}\lambda(\mathbf{r}_i^i)\sum_{j=1}^{i}\tau_{ij}\sigma_{\theta j}\nu_j\dot{q}_j \\
& \quad = -\tau_{0i}\lambda(\omega_i^i)\lambda(\mathbf{r}_i^i)\sum_{j=1}^{i}\tau_{ij}\sigma_{\theta j}\nu_j\dot{q}_j, \\
& -\lambda(\tau_{0i}\mathbf{r}_i^i)\sum_{j=1}^{i}\dot{\tau}_{0j}\sigma_{\theta j}\nu_j\dot{q}_j \\
& \quad = -\lambda(\tau_{0i}\mathbf{r}_i^i)\sum_{j=1}^{i}\tau_{0j}\lambda(\omega_j^j)\sigma_{\theta j}\nu_j\dot{q}_j, \\
& -\sum_{j=1}^{i}\dot{\tau}_{0j}\lambda(\mathbf{l}_j^j)\sum_{m=1}^{j}\tau_{jm}\sigma_{\theta m}\nu_m\dot{q}_m \\
& \quad = -\sum_{j=1}^{i}\tau_{0j}\lambda(\omega_j^j)\lambda(\mathbf{l}_j^j)\sum_{m=1}^{j}\tau_{jm}\sigma_{\theta m}\nu_m\dot{q}_m, \\
& -\sum_{j=1}^{i}\tau_{0j}\lambda(\dot{\mathbf{l}}_j^j)\sum_{m=1}^{j}\tau_{jm}\sigma_{\theta m}\nu_m\dot{q}_m
\end{aligned}$$

$$
\begin{aligned}
&= -\sum_{j=1}^{i} \tau_{0j}\lambda(\sigma_{sj}\nu_j\dot{q}_j)\sum_{m=1}^{j}\tau_{jm}\sigma_{\theta m}\nu_m\dot{q}_m \\
&= \sum_{j=1}^{i}\tau_{0j}\lambda\Big(\sum_{m=1}^{j}\tau_{jm}\sigma_{\theta m}\nu_m\dot{q}_m\Big)\sigma_{sj}\nu_j\dot{q}_j \\
&= \sum_{j=1}^{i}\tau_{0j}\lambda(\omega_j^j)\,\sigma_{sj}\nu_j\dot{q}_j, \\
-\sum_{j=1}^{i}\lambda(\tau_{0j}\mathbf{l}_j^j)\sum_{m=1}^{j}\tau_{0m}\sigma_{\theta m}\nu_m\dot{q}_m &= -\tau_{0j}\sum_{j=1}^{i}\tau_{ij}\lambda(\mathbf{l}_j^j)\,\tau_{j0}\sum_{m=1}^{j}\tau_{0m}\lambda(\omega_m^m)\,\sigma_{\theta m}\nu_m\dot{q}_m, \qquad (2.69)\\
\sum_{j=1}^{i}\tau_{0j}\sigma_{sj}\nu_j\dot{q}_j &= \sum_{j=1}^{i}\tau_{0j}\lambda(\omega_j^j)\,\sigma_{sj}\nu_j\dot{q}_j \\
&= \tau_{0i}\sum_{j=1}^{i}\tau_{ij}\lambda(\omega_j^j)\,\sigma_{sj}\nu_j\dot{q}_j.
\end{aligned}
$$

Substituting the expressions of (2.69) into (2.68) gives the inertial acceleration of a point on the ith link

$$
\begin{aligned}
\mathbf{w}_i^0 = &-\tau_{0i}\Big[\lambda(\mathbf{r}_i^i)\sum_{j=1}^{i}\tau_{ij}\sigma_{\theta j}\nu_j\ddot{q}_j \\
&+\sum_{j=1}^{i}\tau_{ij}\lambda(\mathbf{l}_j^j)\sum_{m=1}^{j}\tau_{jm}\sigma_{\theta m}\nu_m\ddot{q}_m \\
&-\sum_{j=1}^{i}\tau_{ij}\sigma_{sj}\nu_j\ddot{q}_j\Big]-\tau_{0i}\Big[\lambda(\mathbf{r}_i^i)\sum_{j=1}^{i}\tau_{ij}\lambda(\omega_j^j)\,\sigma_{\theta j}\nu_j\dot{q}_j \\
&+\sum_{j=1}^{i}\tau_{ij}\lambda(\mathbf{l}_j^j)\sum_{m=1}^{j}\tau_{im}\lambda(\omega_m^m)\,\sigma_{\theta m}\nu_m\dot{q}_m \\
&+\lambda(\omega_i^i)\,\lambda(\mathbf{r}_i^i)\sum_{j=1}^{i}\tau_{ij}\sigma_{\theta j}\nu_j\dot{q}_j \\
&+\sum_{j=1}^{i}\tau_{ij}\lambda(\omega_j^j)\,\lambda(\mathbf{l}_j^j)\sum_{m=1}^{j}\tau_{jm}\sigma_{\theta m}\nu_m\dot{q}_m \\
&-2\sum_{j=1}^{i}\tau_{ij}\lambda(\omega_j^j)\,\sigma_{sj}\nu_j\dot{q}_j, \qquad (2.70)
\end{aligned}
$$

or with respect to link-tied coordinate systems

$$\begin{aligned}
\mathbf{w}_i^i = &-[\lambda(\mathbf{r}_i^i)\tau_{ij}\sigma_{\theta j}\nu_j\ddot{q}_j \\
&-\sum_{j=1}^{i}\tau_{ij}\lambda(\mathbf{l}_j^j)\sum_{m=1}^{j}\tau_{jm}\sigma_{\theta m}\nu_m\ddot{q}_m \\
&-\sum_{j=1}^{i}\tau_{ij}\sigma_{sj}\nu_j\ddot{q}_j] - [\lambda(\mathbf{r}_i^i)\sum_{j=1}^{i}\tau_{ij}\lambda(\omega_j^j)\sigma_{\theta j}\nu_j\dot{q}_j \\
&+\sum_{j=1}^{i}\tau_{ij}\lambda(\mathbf{l}_j^j)\sum_{m=1}^{j}\tau_{jm}\lambda(\omega_m^m)\sigma_{\theta m}\nu_m\dot{q}_m \\
&+\lambda(\omega_i^i)\lambda(\mathbf{r}_i^i)\sum_{j=1}^{i}\tau_{ij}\sigma_{\theta j}\nu_j\dot{q}_j \\
&+\sum_{j=1}^{i}\tau_{ij}\lambda(\omega_j^j)\lambda(\mathbf{l}_j^j)\sum_{m=1}^{j}\tau_{ij}\sigma_{\theta m}\nu_m\dot{q}_m \\
&+2\sum_{j=1}^{i}\tau_{ij}\lambda(\omega_j^j)\sigma_{sj}\nu_j\dot{q}_j]. \qquad (2.71)
\end{aligned}$$

The vector $\mathbf{l}_j^j = [a_j S_j \sin\alpha_j S_j \cos\alpha_j]^T$ in the above expressions is a constant for revolute pairs, but depends on a phase coordinate $q_j = S_j$ and varies in computations for prismatic and screw pairs. With the linkage parameters and the values of the generalized coordinates q_i and their first and second derivatives $\dot{q}_i$ and $\ddot{q}_i$ the derived expressions yield the linear and angular accelerations of the points and links of robot arm, i.e. are capable of describing the motion of the mechanism. Since these computations are fairly time consuming they are mainly performed on computer.

2.4 PARTITIONED MATRIX REPRESENTATION OF MAIN KINEMATIC RELATIONSHIPS

The kinematic relations derived above can be presented in a more compact form by means of partitioned matrix algebra. This form would be more illustrative and more time efficient in implementing simulating algorithms and analysing the robot arm dynamics on computer.

The partitioned (block) vectors we will use in the following have a dimension of 3-by-1 and will be written in the form $\mathbf{a} = [\mathbf{a}_1^T \mid \mathbf{a}_2^T \mid \mathbf{a}_N^T]^T$ where each element is an ordinary 3×1 position vector. Examples of the block vectors may be the aforementioned vectors of absolute and relative angular velocities in the inertial frame of

reference

$$\tilde{\boldsymbol{\omega}}^0 = [\tilde{\boldsymbol{\omega}}_1^{0T} \mid \tilde{\boldsymbol{\omega}}_2^{0T} \mid \ldots \mid \tilde{\boldsymbol{\omega}}_N^{0T}]^T, \tag{2.72}$$

$$\boldsymbol{\omega}^0 = [\boldsymbol{\omega}_1^{0T} \mid \boldsymbol{\omega}_2^{0T} \mid \quad \mid \boldsymbol{\omega}_N^{0T}]^T. \tag{2.73}$$

Partitioned matrices are composed of ordinary submatrices, namely

$$\mathbf{B} = \left|\begin{array}{c|c|c|c} \mathbf{B}_{11} & \mathbf{B}_{12} & & \mathbf{B}_{1N} \\ \hline \mathbf{B}_{21} & \mathbf{B}_{22} & & \mathbf{B}_{2N} \\ \hline & & & \\ \hline \mathbf{B}_{N1} & \mathbf{B}_{N2} & & \mathbf{B}_{NN} \end{array}\right|. \tag{2.74}$$

If all partitions $\mathbf{B}_{ij}$ are of dimension 3×3 then partitioned matrix $\mathbf{B}$ is an array $3N$-by-$3N$. An example of partitioned matrix is a $3N \times 3N$ matrix $\mathbf{Z}$ having 3×3 unit submatrices on its principal diagonal and below it and 3×3 zero matrices above it

$$\mathbf{Z} = \left|\begin{array}{c|c|c|c} \mathbf{I}_3 & \mathbf{0} & & \mathbf{0} \\ \hline \mathbf{I}_3 & \mathbf{I}_3 & & \mathbf{0} \\ \hline & & & \\ \hline \mathbf{I}_3 & \mathbf{I}_3 & & \mathbf{I}_3 \end{array}\right|. \tag{2.75}$$

Consider a diagonal partitioned $3N \times 3N$ matrix whose partitions are 3×3 submatrices $\boldsymbol{\lambda}(\mathbf{a}_i)$ discussed above

$$\boldsymbol{\Lambda}(\mathbf{a}) = \operatorname{diag}[\boldsymbol{\lambda}(\mathbf{a}_1) \mid \boldsymbol{\lambda}(\mathbf{a}_2) \mid \ldots \mid \boldsymbol{\lambda}(\mathbf{a}_N)]. \tag{2.76}$$

The matrix so defined has the properties similar to that of $\boldsymbol{\lambda}(a_i)$. We will elucidate these properties for block vectors $\mathbf{a}$ and $\mathbf{b}$.

$$\boldsymbol{\Lambda}(\mathbf{a})\,\mathbf{a} = 0, \tag{2.77}$$

$$\boldsymbol{\Lambda}(\mathbf{a})\,\mathbf{b} = -\boldsymbol{\Lambda}(\mathbf{b})\,\mathbf{a}, \tag{2.78}$$

$$\frac{d}{dt}\boldsymbol{\Lambda}(\mathbf{a}) = \boldsymbol{\Lambda}\left(\frac{d\mathbf{a}}{dt}\right), \tag{2.79}$$

$$\boldsymbol{\Lambda}\left(\sum_{k=1}^{N} \mathbf{a}_k\right) = \sum_{k=1}^{N} \boldsymbol{\Lambda}(\mathbf{a}_k), \tag{2.80}$$

$$\boldsymbol{\Lambda}[\boldsymbol{\Lambda}(\mathbf{a})\,\mathbf{b}] = \boldsymbol{\Lambda}(\mathbf{a})\,\boldsymbol{\Lambda}(\mathbf{b}) - \boldsymbol{\Lambda}(\mathbf{b})\,\boldsymbol{\Lambda}(\mathbf{a}). \tag{2.81}$$

It is an easy matter to verify that these equations hold true.

Introduce a $3N \times 3N$ partitioned matrix $\boldsymbol{\tau}$ whose blocks below the principal diagonal are 3×3 submatrices $\boldsymbol{\tau}_{ij}$, defined by the equations (2.4) and (2.12). The main diagonal is composed of 3×3 identity

submatrices, and blocks above it consist of zeroes

$$\tau = \left|\begin{array}{c|c|c|c|c} \mathbf{I} & \mathbf{0} & \mathbf{0} & & \mathbf{0} \\ \hline \tau_{21} & \mathbf{I} & \mathbf{0} & & \mathbf{0} \\ \hline \tau_{31} & \tau_{32} & \mathbf{I} & & \mathbf{0} \\ \hline \vdots & \vdots & \vdots & & \vdots \\ \hline \tau_{N1} & \tau_{N2} & \tau_{N3} & & \mathbf{I} \end{array}\right| \tag{2.82}$$

Note that the submatrices τ and τ_0 are related by

$$\tau = \tau_0 Z \tau_0^T, \tag{2.83}$$

where Z is found from (2.75).

Multiplication of partitioned matrices and column matrices is performed by common rules. With this in mind it can be easily verified that the equality given below is equivalent to (2.54) and (2.47)

$$\omega = \tau\tilde{\omega}. \tag{2.84}$$

Here ω, $\tilde{\omega}$ and τ are defined by the equations (2.73), (2.72) and (2.82).

It may also be demonstrated that the inverse of τ is also of size $3N \times 3N$, namely

$$\tau^{-1} = \left|\begin{array}{c|c|c|c|c|c} \mathbf{I} & \mathbf{0} & \mathbf{0} & & \mathbf{0} & \mathbf{0} \\ \hline -\tau_{21} & \mathbf{I} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & -\tau_{32} & \mathbf{I} & \dots & \mathbf{0} & \mathbf{0} \\ \hline \cdot & & \cdot & & \cdot & \cdot \\ \cdot & \cdot & \cdot & & \cdot & \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} & & -\tau_{N,N-1} & \mathbf{I} \end{array}\right|. \tag{2.85}$$

The principal diagonal of the inverse matrix, τ^{-1}, contains 3×3 unit submatrices; the first diagonal, immediately below the principal one, 3×3 submatrices $-\tau_{21}$, $-\tau_{32}$, ..., $-\tau_{N,N-1}$; and the rest blocks are 3×3 zero submatrices.

If a is a block vector whose elements are defined with respect to link-tied frames, then the following equality holds

$$\begin{aligned} \tau\Lambda(\mathbf{a}) &= \Lambda(\tau\mathbf{a})\tau - \tau\Lambda(\mathbf{a})\tau + \tau\Lambda(\mathbf{a}) \\ &= \Lambda(\tau\mathbf{a})\tau - \tau\Lambda(\mathbf{a})(\tau - \mathbf{I}). \end{aligned} \tag{2.86}$$

To differ ordinary vectors defined in link-tied frames from the block vectors referred to the inertial frame of reference we will mark

the latter with a zero superscript. The block $3N \times 1$ vectors

$$\omega = [\omega_1^{1T} \mid \omega_2^{2T} \mid \quad \mid \omega_N^{NT}]^T, \quad \omega^0 = [\omega_1^{0T} \mid \omega_2^{0T} \mid \ldots \mid \omega_N^{0T}]^T$$

can be related with the help of the $3N \times 3N$ partitioned diagonal matrix

$$\tau_0 = \text{diag}\,[\tau_{10} \mid \tau_{20} \mid \ldots \mid \tau_{N0}] \tag{2.87}$$

as follows

$$\omega = \tau_0 \omega^0. \tag{2.88}$$

One can easily verify that the following holds true

$$\Lambda(\tau_0 a^0) = \tau_0 \Lambda(a^0) \tau_0^T. \tag{2.89}$$

Employing the partitioned matrix algebra we can now put down all the above derived kinematic relations in a more compact form. Two of these relations have already assumed this form in (2.84) and (2.88).

Let us introduce $3N \times 1$ block vectors

$$r^0 = \begin{vmatrix} r_1^0 \\ \hline r_2^0 \\ \hline \\ \hline r_N^0 \end{vmatrix}, \quad l^0 = \begin{vmatrix} l_1^0 \\ \hline l_2^0 \\ \hline \\ \hline l_N^0 \end{vmatrix}, \quad \tilde{v}^0 = \begin{vmatrix} v_1^0 \\ \hline v_2^0 \\ \hline \\ \hline v_N^0 \end{vmatrix}, \quad \omega^0 = \begin{vmatrix} \omega_1^0 \\ \hline \omega_2^0 \\ \hline \\ \hline \omega_N^0 \end{vmatrix},$$

$$r = \begin{vmatrix} r_1^1 \\ \hline r_2^2 \\ \hline \\ \hline r_N^N \end{vmatrix}, \quad l = \begin{vmatrix} l_1^1 \\ \hline l_2^2 \\ \hline \\ \hline l_N^N \end{vmatrix}, \quad v = \begin{vmatrix} v_1^1 \\ \hline v_2^2 \\ \hline \\ \hline v_N^N \end{vmatrix}, \quad \omega = \begin{vmatrix} \omega_1^1 \\ \hline \omega_1^2 \\ \hline \\ \hline \omega_1^N \end{vmatrix}, \tag{2.90}$$

a partitioned $3N \times N$ matrix

$$\nu = \text{diag}\,[\nu_1 \mid \nu_2 \mid \quad \mid \nu_N], \tag{2.91}$$

ordinary $N \times N$ matrices

$$\sigma_\theta = \text{diag}\,[\sigma_{\theta 1} \sigma_{\theta 2} \ldots \sigma_{\theta N}], \tag{2.92}$$

$$\sigma_s = \text{diag}\,[\sigma_{s1} \sigma_{s2} \ldots \sigma_{sN}], \tag{2.93}$$

and an ordinary vector

$$q = [q_1 q_2 \ldots q_N]^T. \tag{2.94}$$

With the introduced set the expressions (2.47) and (2.49) can be rewritten in the form

$$\tilde{\omega} = \nu\sigma_\theta \dot{q}, \tag{2.95}$$

$$\tilde{v} = \nu\sigma_s \dot{q}, \tag{2.96}$$

and for (2.53) and (2.54) we then obtain

$$\omega_0 = \tau_0^T \tau\nu\sigma_\theta \dot{q}, \tag{2.97}$$

$$\omega = \tau\nu\sigma_\theta \dot{q}. \tag{2.98}$$

Denoting

$$\Lambda(r) = R, \ \Lambda(\omega) = \Omega, \ \Lambda(\tilde{\omega}) = \tilde{\Omega}, \ \Lambda(l) = L, \tag{2.99}$$

the equations (2.60), (2.61), (2.66) and (2.67) take on the form

$$v^0 = \tau_0^T [\tau\nu\sigma_s \dot{q} - (R + \tau L)\tau\nu\sigma_\theta \dot{q}], \tag{2.100}$$

$$v = \tau\nu\sigma_s \dot{q} - (R + \tau L)\tau\nu\sigma_\theta \dot{q}, \tag{2.101}$$

$$\varepsilon^0 = \tau_0^T [\tau\nu\sigma_\theta \ddot{q} + \tau\Omega\nu\sigma_\theta \dot{q}], \tag{2.102}$$

$$\varepsilon = \tau\nu\sigma_\theta \ddot{q} + \tau\Omega\nu\sigma_\theta \dot{q}. \tag{2.103}$$

The expressions for the accelerations (2.70) and (2.71) converge to

$$\begin{aligned} w^0 = -\tau_0^T \{ & [(\tau L + R)\tau\nu\sigma_\theta - \tau\nu\sigma_s] \ddot{q} \\ & - [(\tau L + R)\tau\Omega\nu\sigma_\theta \\ & + (\Omega R + \tau\Omega L)\tau\nu\sigma_\theta - 2\tau\Omega\nu\sigma_s] \dot{q} \} \end{aligned} \tag{2.104}$$

$$\begin{aligned} w = - & [(\tau + R)\tau\nu\sigma_\theta - \tau\nu\sigma_s] \ddot{q} \\ & + [-(\tau L + R)\tau\Omega\nu\sigma_\theta \\ & - (\Omega R + \tau\Omega L)\tau\nu\sigma_\theta + 2\tau\Omega\nu\sigma_s] \dot{q}. \end{aligned} \tag{2.105}$$

From (2.95) and (2.78) it follows that

$$\Omega\nu\sigma_s \dot{q} = - \tilde{\Omega}\nu\sigma_s \dot{q},$$

hence the equations (2.104) and (2.105) can be rewritten in the form

$$\begin{aligned} w^0 = -\tau_0^T \{ & [(\tau L + R)\tau\nu\sigma_\theta - \tau\nu\sigma_s] \ddot{q} \\ & + [(\tau L + R)\tau\Omega\tau\nu\sigma_\theta \\ & - (\Omega R + \tau\Omega L)\tau\nu\sigma_\theta + 2\tau\Omega\nu\sigma_s] \dot{q} \}, \end{aligned} \tag{2.106}$$

$$\mathbf{w} = -[(\tau\mathbf{L} + \mathbf{R})\tau\nu\sigma_\theta - \tau\nu\sigma_s]\ddot{\mathbf{q}}$$
$$+ [(\tau\mathbf{L} + \mathbf{R})\tau\Omega\tau\nu\sigma_\theta - (\Omega\mathbf{R} + \tau\Omega\mathbf{L})\tau\nu\sigma_\theta + 2\tau\Omega\nu\sigma_s]\dot{\mathbf{q}}. \quad (2.107)$$

From (2.51) observing partitioned matrix rules we have

$$\dot{\mathbf{l}} = \nu\sigma_s\dot{\mathbf{q}}, \quad (2.108)$$

and (2.14) yields

$$\mathbf{r}^0 = \tau_0^T(\mathbf{r} + \tau\mathbf{l}), \quad (2.109)$$

$$\mathbf{r} = \mathbf{r} + \tau\mathbf{l}. \quad (2.110)$$

The expressions for the block vectors of velocities and accelerations can be obtained directly by differentiating (2.109) and observing the rules

$$\dot{\tau}_0^T = \tau_0^T\Omega, \quad (2.111)$$

$$\dot{\tau}_0 = -\Omega\tau_0, \quad (2.112)$$

$$\dot{\tau} = -\Omega\tau + \tau\Omega. \quad (2.113)$$

The kinematic relationships written in block vectors and matrices notation have given rise to a package of software rutines developed to perform the necessary computations.

2.5 DYNAMIC EQUATIONS OF ROBOT ARM MECHANISM

In deriving the dynamic equations for the robot arm mechanism we will assume all the links of the mechanism to possess absolute stiffness and kinematic joints to be ideal. In the set of the above derived link-tied coordinate frames the base-tied system of reference will again be the inertial system $(0_0X_0Y_0Z_0)$. Prior to generating the dynamic equations we calculate the principal vector and principal moment of inertia forces for each of the links.

Designating m_i the mass of the ith link, the respective vector of inertia forces $\mathbf{F}_{Ji}$ applied in the mass center of the link will be

$$\mathbf{F}_{Ji} = -m_i w_i. \quad (2.114)$$

For the mechanism as a whole introduce the block principal vector of inertia forces

$$\mathbf{F}_J^0 = [\mathbf{F}_{J1}^0 \mid \mathbf{F}_{J2}^0 \mid \ldots \mid \mathbf{F}_{JN}^0]$$

and the partitioned $3N \times 3N$ matrix of mass

$$\mathbf{m} = \text{diag}\,[m_1\mathbf{I} \mid m_2\mathbf{I} \mid \ldots \mid m_N\mathbf{I}].$$

The block vector of center mass acceleration $\mathbf{w}^0$ can be found by (2.106) from which

$$\begin{aligned}\mathbf{F}_J^0 &= -\mathbf{m}\mathbf{w}^0 \\ &= \mathbf{m}\boldsymbol{\tau}_0^{\mathrm{T}} \{[(\mathbf{R} + \boldsymbol{\tau}\mathbf{L})\boldsymbol{\tau}\mathbf{v}\boldsymbol{\sigma}_\theta \\ &\quad - \boldsymbol{\tau}\mathbf{v}\boldsymbol{\sigma}_s]\ddot{\mathbf{q}} - [(\mathbf{R} + \boldsymbol{\tau}\mathbf{L})\boldsymbol{\tau}\tilde{\boldsymbol{\Omega}} \\ &\quad + (\boldsymbol{\Omega}\mathbf{R} + \boldsymbol{\tau}\boldsymbol{\Omega}\mathbf{L})\boldsymbol{\tau}\mathbf{v}\boldsymbol{\sigma}_\theta + 2\boldsymbol{\tau}\boldsymbol{\Omega}\mathbf{v}\boldsymbol{\sigma}_s]\dot{\mathbf{q}}\}. \end{aligned} \tag{2.115}$$

As the next step we will derive the equations for the principal moments of inertia. We attach to each link of the mechanism a right-hand coordinate system ($0_{ci}X_{ci}Y_{ci}Z_{ci}$ for the ith link) so that its origin coincides with the center of mass of the link and the axes are parallel with those of the local frame ($0_iX_iY_iZ_i$). The position of the center of inertia of the ith link is ρ_i^i. The inertia tensor of the ith link defined in the new reference frame ($0_{ci}X_{ci}Y_{ci}Z_{ci}$) is

$$\mathbf{J}_i^{ci} = \begin{vmatrix} J_{x_{ci}x_{ci}} & -J_{x_{ci}y_{ci}} & -J_{x_{ci}z_{ci}} \\ -J_{y_{ci}x_{ci}} & J_{y_{ci}y_{ci}} & -J_{y_{ci}z_{ci}} \\ -J_{z_{ci}x_{ci}} & -J_{z_{ci}y_{ci}} & J_{z_{ci}z_{ci}} \end{vmatrix}. \tag{2.116}$$

The angular momentum of the ith link in the link-tied frame ($0_{ci}X_{ci}Y_{ci}Z_{ci}$) is

$$\mathcal{G}_i^0 = \mathbf{J}_i^{ic}\boldsymbol{\omega}_i^i. \tag{2.117}$$

The principal moment of inertia forces for the ith link, M_{Ji}, can be defined taking a derivative with respect to time of the angular momentum which in the inertial frame of reference is

$$\mathcal{G}_i^0 = \boldsymbol{\tau}_{0i}\mathcal{G}_i^{ci}.$$

Performing the derivative considering (2.55) and (2.67) yields

$$\begin{aligned}\mathbf{M}_{Ji} &= -\frac{\mathrm{d}}{\mathrm{d}t}(\boldsymbol{\tau}_{0i}\mathcal{G}_i^{ci}) = -\dot{\boldsymbol{\tau}}_{0i}\mathbf{J}_i^{ci}\boldsymbol{\omega}_i^i - \boldsymbol{\tau}_{0i}\mathbf{J}_i^{ci}\dot{\boldsymbol{\omega}}_i^i \\ &= -\boldsymbol{\tau}_{0i}\boldsymbol{\lambda}(\boldsymbol{\omega}_i^i)\mathbf{J}_i^{ci}\boldsymbol{\omega}_i^i - \boldsymbol{\tau}_{0i}\mathbf{J}_i^{ci}\boldsymbol{\varepsilon}_i^i, \quad i = 1, 2, \ldots, N. \end{aligned} \tag{2.118}$$

Introduce the block $3N \times 1$ vector of principal moments of the link inertia forces and $3N \times 3N$ matrix of inertia tensors

$$\begin{aligned}&\mathbf{M}_J^0 = [\mathbf{M}_{J1}^{0\mathrm{T}} \mid \mathbf{M}_{J2}^{0\mathrm{T}} \mid \ldots \mid \mathbf{M}_{JN}^{0\mathrm{T}}], \\ &\mathbf{J} = \mathrm{diag}\,[\mathbf{J}_1^{c1} \mid \mathbf{J}_2^{c2} \mid \ldots \mid \mathbf{J}_N^{cN}], \end{aligned} \tag{2.119}$$

then (2.118) can be reduced to

$$\mathbf{M}_J^0 = -\boldsymbol{\tau}_0^{\mathrm{T}}\boldsymbol{\Omega}\mathbf{J}\boldsymbol{\omega} - \boldsymbol{\tau}_0^{\mathrm{T}}\mathbf{J}\boldsymbol{\varepsilon}. \tag{2.120}$$

Using the earlier derived expressions (2.98) and (2.103) for partitioned vectors $\mathbf{w}$ and $\boldsymbol{\varepsilon}$, this equation can be rewritten as

$$\mathbf{M}_j^0 = -\tau_0^T \Omega J \tau \nu \sigma_\theta \dot{\mathbf{q}} + \tau_0^T J \tau \tilde{\Omega} \tau \nu \sigma_\theta \dot{\mathbf{q}} - \tau_0^T J \tau \nu \sigma_\theta \ddot{\mathbf{q}}. \tag{2.121}$$

Besides the forces of inertia, the field of forces acting on each link includes the force of gravity $\mathbf{G}_i$ and external forces $\mathbf{F}_{ei}$ and moments $\mathbf{M}_{ei}$, all of which we shall consider as being applied in the mass center of the link. The block $3N \times 1$ vectors of external forces and moments and gravity forces are

$$\begin{aligned} \mathbf{F}_e^0 &= [\mathbf{F}_{e1}^{0T} \vdots \mathbf{F}_{e2}^{0T} \vdots \quad \vdots \mathbf{F}_{eN}^{0T}]^T, \\ \mathbf{M}_e^0 &= [\mathbf{M}_{e1}^{0T} \vdots \mathbf{M}_{e2}^{0T} \vdots \ldots \vdots \mathbf{M}_{eN}^{0T}]^T, \\ \mathbf{G}^0 &= [\mathbf{G}_1^{0T} \vdots \mathbf{G}_2^{0T} \vdots \quad \vdots \mathbf{G}_N^{0T}]^T. \end{aligned} \tag{2.122}$$

Recall that ν is a unit vector directed along the Z_{i-1} axis and defined with respect to the ith frame of reference.

The forces and moments of reaction acting in the kinematic pairs we denote $\mathbf{F}_{Ri}$ and $\mathbf{M}_{Ri}$ and represent as the partitioned vectors

$$\begin{aligned} \mathbf{F}_R^0 &= [\mathbf{F}_{R1}^{0T} \vdots \mathbf{F}_{R2}^{0T} \vdots \quad \vdots \mathbf{F}_{RN}^{0T}]^T, \\ \mathbf{M}_R^0 &= [\mathbf{M}_{R1}^{0T} \vdots \mathbf{M}_{R2}^{0T} \vdots \quad \vdots \mathbf{M}_{RN}^{0T}]^T \end{aligned} \tag{2.123}$$

Consider a part of linkage composed by links $i, i+1, \ldots, N$, as shown in Fig. 2.4. The influence of the residual section of the linkage we represent by the reaction force $\mathbf{F}_{Ri}^0$ and moment $\mathbf{M}_{Ri}^0$ and a translational force $\mathbf{P}_i^0$ for a prismatic pair or a moment $\mathbf{Q}_i^0$ for a revolute pair. If the joint is a screw pair then $\mathbf{Q}_i^0$ and $\mathbf{P}_i^0$ represent this action together. In accordance with d'Alembert's principle [51] the sum of all forces, including inertial, and moments acting on a mechanical system must be equal to zero. So we will write the respective balance equation with respect to the inertial (zeroth) frame of reference for the forces and reactions produced by actuators in the $i+1, i+2, \ldots$ N kinematic pairs

$$\sum_{k=i}^{N} (\mathbf{G}_k^0 + \mathbf{F}_{ek}^0 + \mathbf{F}_{Jk}^0) + \mathbf{F}_{Ri}^0 + \mathbf{P}_{i0} = 0, \quad i = 1, 2, \ldots, N. \tag{2.124}$$

The equation of moments about the point O_{i-1} written in the inertial system has the form

$$\begin{aligned} &\sum_{k=i}^{N} [(\mathbf{M}_{ek}^0 + \mathbf{M}_{Jk}^0) + \lambda (\sum_{j=i}^{k} \mathbf{l}_j^0 + \mathbf{r}_k^0)(\mathbf{F}_{ek}^0 + \mathbf{G}_k^0 + \mathbf{F}_{Jk}^0)] \\ &\quad + \mathbf{M}_{Ri}^0 + \mathbf{Q}_i^0 = 0, \quad i = 1, 2, \ldots, N, \end{aligned} \tag{2.125}$$

here

$$\left(\sum_{j=i}^{k} \mathbf{l}_j^0 + \mathbf{r}_k^0\right)(\mathbf{F}_{ek}^0 + \mathbf{G}_k^0 + \mathbf{F}_{Jk}^0) = \left(\sum_{j=1}^{k} \mathbf{l}_j^0 + \mathbf{r}_k^0\right) \times (\mathbf{F}_{ek}^0 + \mathbf{G}_k^0 + \mathbf{F}_{Jk}^0) \tag{2.126}$$

enote the moments of the forces $\mathbf{F}_{ek}$, $\mathbf{G}_k$ and $\mathbf{F}_{Jk}$ about the point $_{i-1}$. Recall that $\mathbf{l}_j$ is the vector connecting the local origins O_{j-1} nd O_j, and $\mathbf{r}_k$ the vector from O_k to C_k, the mass center of the kth ink.

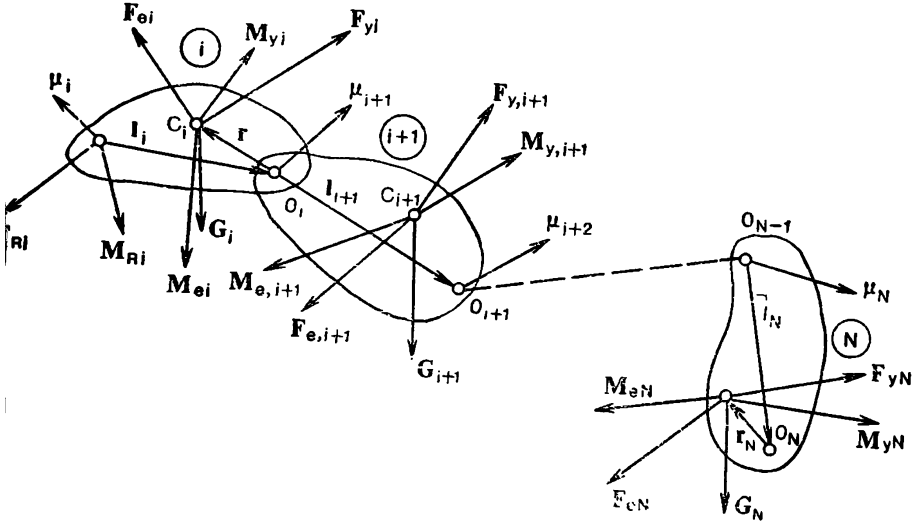


Fig. 2.4 Forces and moments acting on manipulator links

Changing the order of summation in the dual sum of (2.125) gives

$$\sum_{=}^{N} (\mathbf{M}_{ek}^0 + \mathbf{M}_{Jk}^0) + \sum_{k=i}^{N} \lambda(\mathbf{r}_k^0)(\mathbf{F}_{ek}^0 + \mathbf{G}_k^0 + \mathbf{F}_{Jk}^0)$$

$$+ \sum_{j=1}^{N} \sum_{k=i}^{N} \lambda(\mathbf{l}_j^0)(\mathbf{F}_{ek}^0 + \mathbf{G}_k^0 + \mathbf{F}_{Jk}^0) + \mathbf{M}_{Ri}^0 + \mathbf{Q}_i^0 = 0, \quad i = 1, \qquad N \tag{2.127}$$

hich upon substitution for vector product of two vectors defined ith respect to link-tied coordinate systems yields

$$\sum_{=i}^{N} (\mathbf{M}_{ek}^0 + \mathbf{M}_{Jk}^0) + \sum_{k=i}^{N} \tau_{0k}\lambda(\mathbf{r}_k^k)\tau_{k0}(\mathbf{F}_{ek}^0 + \mathbf{G}_k^0 + \mathbf{F}_{Jk}^0)$$

$$+ \sum_{j=1}^{N} [\tau_{0j}\lambda(\mathbf{l}_j^j)\tau_{j0} \sum_{k=i}^{N} (\mathbf{F}_{ek}^0 + \mathbf{G}_k^0 + \mathbf{F}_{Jk}^0) + \mathbf{M}_{Ri}^0 + \mathbf{Q}_i^0 = 0,$$

$$i = 1, 2, \ldots, N. \tag{2.128}$$

To obtain the projection of (2.128) on the Z_{i-1} axis of the ith kinematic pair it should be multiplied by the unit vector $\mathbf{e}_i^{0T}$ of this axis; the result is

$$\mathbf{e}_{i-1}^{0T}\Big\{\sum_{k=1}^{N}(\mathbf{M}_{ek}^0+\mathbf{M}_{Jk}^0)+\sum_{k=i}^{N}\tau_{0k}\lambda\,\mathbf{r}_k^k)\,\tau_{k0}\;(\mathbf{F}_{ek}^0+\mathbf{G}_k^0+\mathbf{F}_{Jk}^0)$$

$$+\sum_{j=1}^{N}\tau_{0j}\lambda\,(\mathbf{l}_j^j)\,\tau_{j0}\sum_{k=i}^{N}(\mathbf{F}_{ek}^0+\mathbf{G}_k^0+\mathbf{F}_{Jk}^0)\Big\}+\mu_i=0,$$

$$\mu_i=\mathbf{l}_{i0}^{T}\mathbf{Q}_i^0,\qquad i=1,\ 2,\ \ldots,\ N. \tag{2.129}$$

If the ith kinematic pair, coupling the $(i-1)$ st and ith links, is prismatic then the projection of force on the Z_{i-1} axis becomes zero, thus for the projection of (2.124) on this axis we have

$$\mathbf{e}_{i-1}^{0T}\sum_{k=i}^{N}(\mathbf{G}_k^0+\mathbf{F}_{ek}^0+\mathbf{F}_{Jk}^0)+\mu_i=0, \tag{2.130}$$

$$\mu_i=\mathbf{e}_{i-1}^{0T}\mathbf{P}_i^0,\qquad i=1,\ 2,\qquad N.$$

If the pair is screw, both the projection of the reaction force $\mathbf{F}_{Ri}^0$ and the projection of its moment $\mathbf{M}_{Ri}^0$ on Z_{i-1} vanish, but μ_i is determined as

$$\mu_i=\mathbf{e}_{i-1}^{0T}\mathbf{Q}_i^0\sigma_{\theta i}+\mathbf{e}_{i-1}^{0T}\mathbf{P}_i^0\sigma_{si},\qquad i=1,\ 2,\ \ldots,\ N. \tag{2.131}$$

Summing up the projections of (2.124) and (2.128) on Z_{i-1}, multiplying them, respectively, on σ_{si} and $\sigma_{\theta i}$ and observing (2.131) we obtain for the screw pair

$$\mathbf{e}_{i-1}^{0T}\Big[\sigma_{si}\sum_{k=i}^{N}(\mathbf{G}_k^0+\mathbf{F}_{ek}^0+\mathbf{F}_{Jk}^0)$$

$$+\sigma_{\theta i}\Big[\sum_{k=i}^{N}(\mathbf{M}_{ek}^0+\mathbf{M}_{ik}^0)+\sum_{k=i}^{N}\tau_{0k}\lambda\,(\mathbf{r}_k^k)\,\tau_{k0}\,(\mathbf{F}_{ek}^0+\mathbf{G}_k^0+\mathbf{F}_{Jk}^0)$$

$$+\sum_{j=1}^{N}\tau_{0j}\lambda\,(\mathbf{l}_j^j)\,\tau_{j0}\sum_{k=i}^{N}(\mathbf{F}_{ek}^0+\mathbf{G}_k^0+\mathbf{F}_{Jk}^0)\Big]\Big\}+\mu_i=0,$$

$$\mu_i=\mathbf{e}_{i-1}^{0T}(\sigma_{\theta i}\mathbf{Q}_i^0+\sigma_{si}\mathbf{P}_i^0),\qquad i=1,\ 2,\ \ldots,\ N. \tag{2.132}$$

Note that μ_i is the stress created by an actuator in the ith kinematic pair; for a prismatic pair this is a force, for a revolute pair this is a moment, and for a screw pair this is either a force or a moment depending on the design of the gear.

Should we recall that the variables $\sigma_{\theta i}$ and σ_{si} take on the values of 1 or 0 according as the type of the kinematic pair

$$\sigma_{Si} = \begin{cases} 1, & \text{if the } i\text{th pair is prismatic} \\ 0, & \text{if the } i\text{th pair is revolute} \end{cases}$$

$$\sigma_{\theta i} = \begin{cases} 1, & \text{if the } i\text{th pair is revolute} \\ 0, & \text{if the } i\text{th pair is prismatic} \end{cases}$$

then it becomes obvious that (2.132) is valid for both (2.129) and (2.130). Hence, in the following we will only discuss the equation (2.132).

Postmultiplying the first multiplier in (2.132) by the identity matrix $\tau_{i0}^T\tau_{i0}$ results in

$$\mathbf{e}_{i-1}^{0T}\tau_{i0}^T\tau_{i0} = \nu_i^T\tau_{i0}$$

where $\nu_i = \tau_{i0}\mathbf{e}_{i-1}^0$ is the unit vector in the Z_{i-1} axis referred to the ith frame (see Fig. 2.1). This manipulation yields

$$\sigma_{si}\nu_i^T\tau_{i0}\sum_{k=i}^{N}(\mathbf{G}_k^0 + \mathbf{F}_{ek}^0 + \mathbf{F}_{Jk}^0) + \sigma_{\theta i}\nu_i\tau_{i0}\Big\{\sum_{k=i}^{N}(\mathbf{M}_{ek}^0 + \mathbf{M}_{Jk}^0)$$
$$+ \sum_{k=i}^{N}\tau_{0k}\lambda(\mathbf{r}_k^k)\tau_{k0}(\mathbf{F}_{ek}^0 + \mathbf{G}_k^0 + \mathbf{F}_{Jk}^0) +$$
$$+ \sum_{j=1}^{N}\tau_{0j}\lambda(\mathbf{l}_j^j)\tau_{j0}\sum_{k=1}^{N}(\mathbf{F}_{ek}^0 + \mathbf{G}_k^0 + \mathbf{F}_{Jk}^0)\Big\} + \mu_i = 0,$$
$$\mu_i = \sigma_{\theta i}\nu_i^T\tau_{i0}\mathbf{Q}_i^0 + \sigma_{si}\nu_i^T\tau_{i0}\mathbf{P}_i^0, \qquad i = 1, 2, \ldots, N. \tag{2.133}$$

Using the partitioned matrices introduced above, the expression (2.133) can be rewritten in a more compact form

$$\sigma_s\nu^T\tau_0\mathbf{Z}^T(\mathbf{G}^0 + \mathbf{F}_e^0 + \mathbf{F}_J^0) + \sigma_\theta\nu^T\tau_0\mathbf{Z}^T[(\mathbf{M}_e^0 + \mathbf{M}_J^0) + \tau_0^T\mathbf{R}\tau_0(\mathbf{F}_e^0 + \mathbf{G}^0 + \mathbf{F}_J^0)$$
$$+ \tau_0^T\mathbf{L}\tau_0\mathbf{Z}^T(\mathbf{F}_e^0 + \mathbf{G}^0 + \mathbf{F}_J^0)] + \mu = 0,$$
$$\mu = [\mu_1\mu_2, \ldots, \mu_N]^T. \tag{2.134}$$

or considering that $\tau_0\mathbf{Z}\tau_0^T = \tau^T$, $\mathbf{R} + \mathbf{L}\tau^T = -(\mathbf{R} + \mathbf{L}\tau)^T$ and $\tau_0^T\tau = \mathbf{I}$, we get

$$[\sigma_s\nu^T\tau^T - \sigma_\theta\nu^T\tau^T(\mathbf{R} + \tau\mathbf{L})^T]\tau_0(\mathbf{G}^0 + \mathbf{F}_e^0) + \sigma_\theta\nu^T\tau^T\tau_0\mathbf{M}_e^0 + \sigma_\theta\nu^T\tau^T\tau_0\mathbf{M}_J^0$$
$$+ [\sigma_s\nu^T\tau^T - \sigma_\theta\nu^T\tau^T(\mathbf{R} + \tau\mathbf{L})^T]\tau_0\mathbf{F}_J^0 + \mu = 0. \tag{2.135}$$

To write the dynamic equation for the robot arm we replace the forces of inertia $\mathbf{F}_J$ and moments of inertia $\mathbf{M}_J$ with their expressions (2.114) and (2.121) in derivatives of generalized coordinates.

This substitution being performed, we get a system of N second order differential equations, which describes the motion of the mechanism under the influence of the forces $\mathbf{F}_e^0$ and $\mathbf{G}^0$ and moments $\mathbf{M}_e^0$, and forces $\boldsymbol{\mu}$ generated by actuators

$$\mathbf{A}(\mathbf{q})\ddot{\mathbf{q}} = \mathbf{B}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{C}(\mathbf{q})\mathbf{M}_e^0 + \mathbf{D}(\mathbf{q})(\mathbf{G}^0 - \mathbf{M}_e^0) - \boldsymbol{\mu}, \tag{2.136}$$

where $\mathbf{A}(\mathbf{q})$, $\mathbf{B}(\mathbf{q}, \dot{\mathbf{q}})$, $\mathbf{C}(\mathbf{q})$, and $\mathbf{D}(\mathbf{q})$ are the N N matrices defined by

$$\begin{aligned}\mathbf{A}(\mathbf{q}) = {} & \sigma_\theta \nu^T \tau^T [\mathbf{J} + (\mathbf{R} + \tau\mathbf{L})^T \mathbf{m} (\mathbf{R} + \tau\mathbf{L})] \tau\nu\sigma_\theta \\ & + \sigma_\theta \nu^T \tau^T (\mathbf{R} + \mathbf{L}\tau^T) \mathbf{m}\tau\nu\sigma_s - \sigma_s \nu^T \tau^T \mathbf{m} (\mathbf{R} + \tau\mathbf{L}) \tau\nu\sigma_\theta \\ & + \sigma_s \nu^T \tau^T \mathbf{m}\tau\nu\sigma_s,\end{aligned} \tag{2.137}$$

$$\begin{aligned}\mathbf{B}(\mathbf{q}, \dot{\mathbf{q}}) = {} & \sigma_\theta \nu^T \tau^T \{[\mathbf{J} + (\mathbf{R} + \tau\mathbf{L})^T \mathbf{m} (\mathbf{R} + \tau\mathbf{L})] \tau\tilde{\Omega} \\ & - [\Omega\mathbf{J} + (\tau\mathbf{L} + \mathbf{R})^T \mathbf{m} (\Omega\mathbf{R} + \tau\Omega\mathbf{L})\} \tau\nu\sigma_\theta \\ & + 2\sigma_\theta \nu^T \tau^T (\mathbf{R} + \tau\mathbf{L})^T \mathbf{m}\tau\Omega\nu\sigma_s - \sigma_s \nu^T \tau^T \mathbf{m} [(\mathbf{R} + \tau\mathbf{L}) \tau\tilde{\Omega} \\ & - (\Omega\mathbf{R} + \tau\Omega\mathbf{L})] \tau\nu\sigma_\theta - 2\sigma_s \nu^T \tau^T \mathbf{m}\tau\Omega\nu\sigma_s,\end{aligned} \tag{2.138}$$

$$\mathbf{C}(\mathbf{q}) = \sigma_\theta \nu^T \tau^T \tau_0 = \sigma_\theta \nu^T \tau_0 \mathbf{Z}^T, \tag{2.139}$$

$$\mathbf{D}(\mathbf{q}) = [\sigma_s \nu^T \tau^T - \sigma_\theta \nu^T \tau^T (\mathbf{R} + \tau\mathbf{L})^T] \tau_0. \tag{2.140}$$

The derived differential equation of motion may be used in both analysis and synthesis of control system governing the robot's arm.

2.6 DYNAMIC EQUATIONS FOR A MOVING BASE ROBOT

In the preceding Section we have derived the differential equations describing the dynamics of the robot mechanism for the case when the frame (base) of the manipulator is fixed and the base-tied coordinate system $(0_0 X_0 Y_0 Z_0)$ is referred to as inertial. Some robots, however, can be moved while operating, or can be made with movable base. We intend to demonstrate that the deduced dynamic equations can be applied to describe the dynamics of the movable robots as well.

The base to which the arm is attached we will believe to be an absolutely rigid body of six degrees of freedom. The coordinate system $(0_6 X_6 Y_6 Z_6)$ is tied with the base so that the axis Z_6 is co-linear with the axis of the kinematic pair connecting the first link of the mechanism to the base. The axis X_6 is along the common perpendicular to the construction line of robot carrier (say, a space probe) and Z_6. The axis Y_6 completes the right-hand orthogonal system. The links of the manipulator linkage will be enumerated starting from the

base: 7, 8, $N + 6$; here N is the number of links in the mechanism.

Introduce in our consideration an imaginary mechanism with six degrees of freedom, which contains the base fixed in it, as it were, see Fig. 2.5. The mechanism consists of six links connected by three sliding and three revolute pairs of connectivity one. The masses

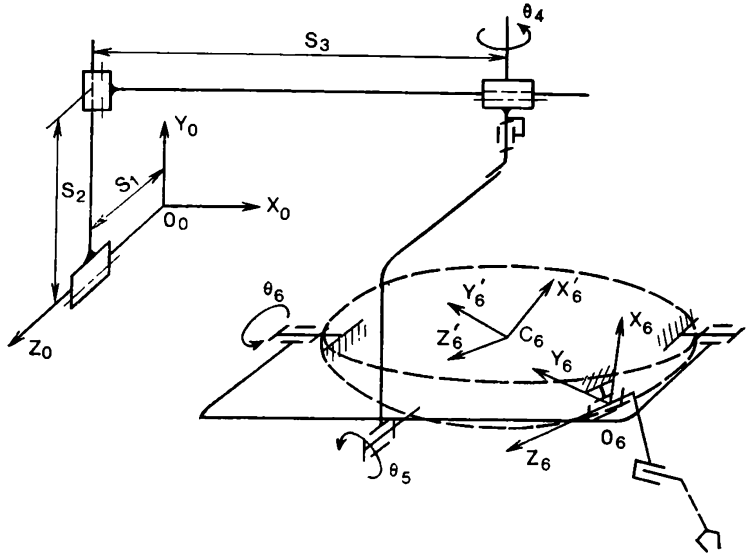


Fig. 2.5 Imaginary mechanism governing the movement of robot's base

and moments of inertia of the mechanism links will be assumed equal to zero. The base is bound to move along the axes X_0, Y_0 and Z_0 of the inertial frame and its rotation will be discussed in a fictitious Hooke's joint whose axes intersect in a point C_6, the mass center of the base. We fix to each link of the mechanism a reference frame in the manner we have done above. Sliding displacements along the axes Z_0, Z_1, and Z_2 we denote, respectively, as, S_1, S_2, and S_3, and the angles of rotation about the axes Z_3, Z_4, and Z_5, respectively, as θ_4, θ_5, and θ_6. These angles are Euler with θ_4 being the yaw angle, θ_5 the pitch angle, and θ_6 the roll angle.

Now we will assume that the fictitious mechanism, the base and the arm of the robot teamed up, compose a single gross linkage of $N + 6$ links connected by kinematic pairs of connectivity one. For this gross mechanism we summarize the values of the parameters defining transformation from one link-tied reference frame to another.

	θ_i		a_i	α_i	l_i		$\sigma_{\theta i}$	σ_{si}
1	0	q_1	0	$-\pi/2$	0	0	0	1
2	$\pi/2$	q_2	0	$\pi/2$	0	0	0	1
3	π	q_3	0	$\pi/2$	0	0	0	1
4	q_4	0	0	$-\pi/2$	0	0	1	0
5	q_5	0	0	$\pi/2$	0	0	1	0
6	q_6	0	a_6	α_6	l_6	$-l_6$	1	0
7	θ_7	S_7	a_7	α_7	l_7	r_7	$\sigma_{\theta 7}$	σ_{s7}
$N+6$	$\dot{\theta}_{N+6}$	$\dot{S}_{N+6}$	a_{N+6}	α_{N+6}	$\dot{l}_{N+6}$	r_{N+6}	$\sigma_{\theta, N+6}$	$\sigma_{s,\ N+6}$

We tied to the base two more coordinate systems, namely, $(C_6X_6'Y_6'Z_6')$ whose origin is in the mass center point C_6 and the respective axes are parallel with X_6, Y_6 and Z_6; and the frame-fixed $(C_6X_fY_fZ_f)$ also with the origin in the center of mass C_6. The axis X_f is directed along the construction line of the carrier, the Y_f axis lies in the plane of symmetry perpendicular to the construction line, and the axis Z_f completes the right-hand orthogonal system.

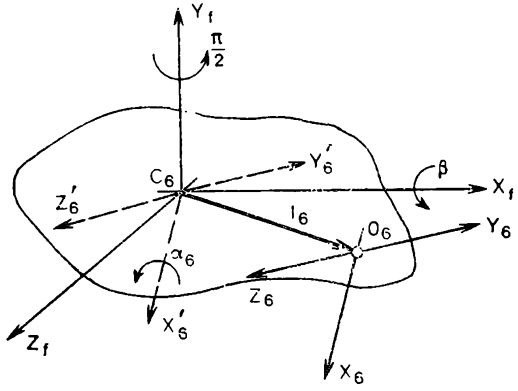


Fig. 2.6 Coordinate systems tied to robot's moving base

The carrier inertia tensor is normally specified in the frame $(C_6X_fY_fZ_f)$. A change from the frame $(C_6X_fY_fZ_f)$ to $(C_6X_6'Y_6'Z_6')$ is made in three rotations, as is depicted in Fig. 2.6. First, around the Y_f axis through an angle $-\pi/2$, then about a new position of Z_f through an angle β such that X_f and X_6' coincide, and, finally, about the X_6' axis through an angle α_6 for the systems to completely coincide. The angles β and α_6 are constant and decided by the geometry of arm

attachment to the base. The transformation matrix T_{6f} has the form

$$T_{6f} = \begin{vmatrix} 0 & \sin\beta & \cos\beta \\ -\sin\alpha_6 & \cos\beta\cos\alpha_6 & -\sin\beta\cos\alpha_6 \\ -\cos\alpha_6 & -\cos\beta\sin\alpha_6 & \sin\beta\sin\alpha_6 \end{vmatrix}.$$

The inertia tensor of the base J_6 with respect to the base-tied coordinate system $(O_6X_6Y_6Z_6)$ can be obtained by resorting to the expression

$$J_6^6 = T_{6f}J_fT_{6f}^T - \lambda(l_6^6)\, m_6\lambda(l_6^6),$$

where l_6^6 is the vector connecting the points O_6 and C_6 and defined in the frame $(O_6X_6Y_6Z_6)$.

According to the principle adopted in constructing the tied reference frames, the generalized coordinates vanish, that is

$$q_1 = S_1 = 0, \quad q_2 = S_2 = 0, \quad \text{and} \quad q_3 = S_3 = 0,$$

when the origin of the inertial frame is taken on in the mass center of the carrier.

The imaginary mechanism under discussion imposes no constraints on how the robot carrier moves (but for the case $q_6 = 0$), has a zero mass, and its kinematic pairs generate no forces or moments.

$$\mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5 = 0,$$

$$G_1 = G_2 = G_3 = G_4 = G_5 = 0,$$

$$M_{e1} = M_{e2} = M_{e3} = M_{e4} = M_{e5} = 0,$$

$$F_{e1} = F_{e2} = F_{e3} = F_{e4} = F_{e5} = 0.$$

Therefore the carrier will move as a free body.

The external forces and moments acting on the carrier as the sixth link of the mechanism are due to the action of the control actuator, the opposing forces, etc. In the general case they depend on the parameters of motion and position of the carrier. With allowance for the suggestions made and designations introduced the dynamic equation for the manipulator (2.136) can be applied to describe the motion of the mechanism for a robot with a moving base.

2.7 LINEARIZING EQUATIONS OF MOTION

Let us linearize the equation (2.136) describing the dynamics of the robot

$$A(q)\ddot{q} = B(q, \dot{q})\,\dot{q} + C(q)\,M_e^g + D(q)(G^0 + F_e^0) + \mu.$$

We take on as a reference curve, defined by a vector of the generalized coordinates q^*, the most frequent path in the manipulator workspace.

In the general case, the coefficients of the linearized equation will depend on the reference values of the generalized coordinates $\mathbf{q}^*(t)$ and their derivatives $\dot{\mathbf{q}}^*(t)$ and $\ddot{\mathbf{q}}^*(t)$, hence, the linearized equations will be nonstationary with variable coefficients. In a particular case of an equilibrium state assumed for the reference values of the generalized coordinates, i.e., $\mathbf{q}^*(t) = \mathbf{q}^*$, $\dot{\mathbf{q}}^*(t) = \ddot{\mathbf{q}}^*(t) = 0$, we obtain a stationary linear differential equation with constant coefficients.

Let us assume that disturbances deviate each of the generalized coordinates $q_i(t)$ from the reference value $q_i^*(t)$ for a quantity $\Delta q_i(t)$ so that for a new state

$$q_i(t) = q_i^*(t) + \Delta q_i(t), \quad i = 1, 2, \ldots, N \tag{2.141}$$

or recast in the vector form

$$\mathbf{q}(t) = \mathbf{q}^*(t) + \Delta\mathbf{q}(t). \tag{2.142}$$

This operation makes the terms of the dynamic equation take on some new values, different of that they had along the nondisturbed curve $\mathbf{q}^*(t)$; the motion of the mechanism will then be described by the equation

$$\begin{aligned}\Delta[\mathbf{A}(\mathbf{q})\ddot{\mathbf{q}}] = {} & \Delta[\mathbf{B}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}] + \Delta[\mathbf{C}(\mathbf{q})\mathbf{F}_e^0] \\ & + \Delta[\mathbf{D}(\mathbf{q})(\mathbf{G}^0 + \mathbf{F}_e^0)] + \Delta\boldsymbol{\mu} - \boldsymbol{\mu}_e,\end{aligned} \tag{2.143}$$

wherein $\boldsymbol{\mu}_e$ designates the vector of disturbances created by external forces and moments, and the forces and moments applied to links in moving the mechanism over the reference curve

$$\begin{aligned}\boldsymbol{\mu}_e = {} & -\mathbf{A}(\mathbf{q}^*)\ddot{\mathbf{q}}^* + \mathbf{B}(\mathbf{q}^*, \dot{\mathbf{q}}^*)\dot{\mathbf{q}}^* \\ & + \mathbf{D}(\mathbf{q}^*)(\mathbf{G}^{*0} + \mathbf{F}^{*0}) + \boldsymbol{\mu}^*.\end{aligned} \tag{2.144}$$

Obviously, for an exact, nondisturbed model $\boldsymbol{\mu}_e = 0$. The increments $\Delta[\mathbf{A}(\mathbf{q})\ddot{\mathbf{q}}]$, $\Delta[\mathbf{B}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}]$, $\Delta[\mathbf{C}(\mathbf{q})\mathbf{M}_e^0]$, and $\Delta[\mathbf{D}(\mathbf{q})(\mathbf{G}^0 + \mathbf{F}_e^0)]$ can be computed by expanding the vectors $\mathbf{A}(\mathbf{q})\ddot{\mathbf{q}}$, $\mathbf{B}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}$, $\mathbf{C}(\mathbf{q})\mathbf{M}_e^0$, and $\mathbf{D}(\mathbf{q})(\mathbf{G}^0 + \mathbf{F}_e^0)$ into their Taylor series

$$\Delta[\mathbf{A}(\mathbf{q})\ddot{\mathbf{q}}] = \mathbf{A}^*\Delta\ddot{\mathbf{q}} + \left.\frac{\partial(\mathbf{A}(\mathbf{q})\ddot{\mathbf{q}}^*)}{\partial\mathbf{q}}\right|_* \Delta\mathbf{q} + \boldsymbol{\alpha}_A,$$

$$\Delta[\mathbf{B}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}] = \mathbf{B}^*\Delta\dot{\mathbf{q}} + \left.\frac{\partial(\mathbf{B}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}^*)}{\partial\mathbf{q}}\right|_* \Delta\mathbf{q} + \left.\frac{\partial(\mathbf{B}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}^*)}{\partial\dot{\mathbf{q}}}\right|_* \Delta\dot{\mathbf{q}} + \boldsymbol{\alpha}_B,$$

$$\Delta[\mathbf{C}(\mathbf{q})\mathbf{M}_e^0] = \mathbf{C}^*\Delta\mathbf{M}_e^0 + \left.\frac{\partial(\mathbf{C}(\mathbf{q})\mathbf{M}_e^0)}{\partial\mathbf{q}}\right|_* \Delta\mathbf{q} + \boldsymbol{\alpha}_C,$$

$$\Delta [D(q)(G^0 + F_e^0)] = D^* (\Delta G^0 + \Delta F_0^e) + \frac{\partial (D(q)(G^0 + F_e^0))}{\partial q}\Big|_* \Delta q + \alpha_D,$$

where $A^* = A(q^*)$, $B^* = B(q^*, \dot{q}^*)$, $C^* = C(q^*)$, $D^* = D(q^*)$; $\frac{\partial(\cdot)}{\partial q}$, $\frac{\partial(\cdot)}{\partial \dot{q}}$ are matrix derivatives with respect to the vectors q and $\dot{q}$ at the undisturbed values $q(t) = q^*(t)$, $\dot{q}(t) = \dot{q}^*(t)$, $\ddot{q}(t) = \ddot{q}^*(t)$, $F_e^0(t) = F_e^{0*}(t)$, $M_e^0 = M_e^{0*}(t)$, $G^0 = G^{0*}$; α_A, α_B, α_C, and α_D are vectors whose norms $(\|\alpha\| = \sqrt{\alpha_1^2 + \ldots + \alpha_N^2})$ decrease faster than the norm of the increment $\|\Delta q\|$, that is

$$\|\alpha_A\| = o(\|\Delta q\|),$$

$$\|\alpha_B\| = o(\|\Delta q\|) + o(\|\Delta \dot{q}\|),$$

$$\|\alpha_C\| = o(\|\Delta q\|),$$

$$\|\alpha_D\| = o(\|\Delta q\|).$$

Introducing the designation

$$\frac{\partial [A(q)\ddot{q}^*]}{\partial q}\Big|_* = A_1, \qquad \frac{\partial [B(q, \dot{q})\dot{q}^*]}{\partial q}\Big|_* = B_1,$$

$$\frac{\partial [B(q, \dot{q})\dot{q}^*]}{\partial q}\Big|_* = B_2, \qquad \frac{\partial [C(q)M_e^{0*}]}{\partial q}\Big|_* = C_1,$$

$$\frac{\partial [D(q)(G^{0*} + F_e^{0*})]}{\partial q} = D_1,$$

we can reduce the equation (2.143) accurate to the terms of the order higher than $\Delta q + \Delta \dot{q}$ to the form

$$a(q^*)\Delta\ddot{q} + b(q^*, \dot{q}^*)\Delta\dot{q} + c(q^*, \dot{q}^*, \ddot{q}^*)\Delta q = \Delta\mu + C^*\Delta M_e^0 + D^*(\Delta F_e^0 + \Delta G^0) - \mu_e, \tag{2.146}$$

where

$$\begin{aligned} a(q^*) &= A^*, \\ b(q^*, \dot{q}^*) &= -B^* - B_2, \\ c(q^*, \dot{q}^*, \ddot{q}^*) &= A_1 - B_1 - C_1 - D_1. \end{aligned} \tag{2.147}$$

For the general case, the matrices A, B, C, and D are defined by the equations (2.136-139). The form of (2.146) for the dynamic equation substantially facilitates the linearization procedure based on the previously introduced matrix formalizm. To illustrate, the coefficients in (2.146) contain only five matrix variables τ_0, τ, Ω, $\tilde{\Omega}$,

and L, therefore to effect the linearization procedure it will suffice deducing the formulas for the derivatives of the products between these matrices and constant vectors.

From (2.48), (2.49), (2.96), (2.99), and (2.105) with $\frac{\partial \mathbf{q}}{\partial \mathbf{q}} = \mathbf{I}$ we have for an arbitrary constant vector $\mathbf{r}$

$$\frac{\partial (\tilde{\Omega}\mathbf{r})}{\partial \dot{\mathbf{q}}} = -\frac{\partial (\Lambda(\mathbf{r})\tilde{\omega})}{\partial \dot{\mathbf{q}}} = -\Lambda(\mathbf{r})\frac{\partial \tilde{\omega}}{\partial \dot{\mathbf{q}}}$$

$$= -\Lambda(\mathbf{r})\frac{\partial (\nu\sigma_\theta \dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}} = -\Lambda(\mathbf{r})\nu\sigma_\theta, \tag{2.148}$$

$$\frac{\partial (\Omega\mathbf{r})}{\partial \dot{\mathbf{q}}} = -\frac{\partial [\Lambda(\mathbf{r})\omega]}{\partial \dot{\mathbf{q}}} = -\Lambda(\mathbf{r})\frac{\partial \omega}{\partial \dot{\mathbf{q}}}$$

$$= -\Lambda(\mathbf{r})\frac{\partial (\tau\nu\sigma_\theta \dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}} = -\Lambda(\mathbf{r})\tau\nu\sigma_\theta. \tag{2.149}$$

To take the derivatives of the products $\mathbf{Lr}$, $\tau_0\mathbf{r}$, and $\tau\mathbf{r}$ we resort to the following relations

$$\frac{\partial (\mathbf{Lr})}{\partial \mathbf{q}} = \frac{\partial (\dot{\mathbf{L}}\mathbf{r})}{\partial \dot{\mathbf{q}}}, \quad \frac{\partial (\tau_0\mathbf{r})}{\partial \mathbf{q}} = \frac{\partial (\dot{\tau}_0\mathbf{r})}{\partial \dot{\mathbf{q}}}, \quad \frac{\partial (\tau\mathbf{r})}{\partial \mathbf{q}} = \frac{\partial (\dot{\tau}\mathbf{r})}{\partial \dot{\mathbf{q}}\,\mathbf{q}}. \tag{2.150}$$

From (2.65) it follows

$$\frac{\partial (\mathbf{Lr})}{\partial \mathbf{q}} = -\Lambda(\mathbf{r})\frac{\partial \mathbf{l}}{\partial \mathbf{q}} = -\Lambda(\mathbf{r})\frac{\partial (\nu\sigma_s \dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}} = -\Lambda(\mathbf{r})\nu\sigma_s. \tag{2.151}$$

and from (2.69)

$$\frac{\partial (\tau_0\mathbf{r})}{\partial \mathbf{q}} = \frac{\partial (\dot{\tau}_0\mathbf{r})}{\partial \dot{\mathbf{q}}} = -\frac{\partial (\Lambda(\omega)\tau_0\mathbf{r})}{\partial \dot{\mathbf{q}}} = \Lambda(\tau_0\mathbf{r})\frac{\partial \omega}{\partial \dot{\mathbf{q}}}$$

$$= \Lambda(\tau_0\mathbf{r})\frac{\partial (\tau\nu\sigma_\theta \dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}} = \Lambda(\tau_0\mathbf{r})\tau\nu\sigma_\theta. \tag{2.152}$$

From (2.113) and (2.98) we get

$$\frac{\partial (\tau\mathbf{r})}{\partial \mathbf{q}} = \frac{\partial (\dot{\tau}\mathbf{r})}{\partial \dot{\mathbf{q}}} = -\frac{\partial (\Omega\tau\mathbf{r})}{\partial \dot{\mathbf{q}}} + \frac{\partial (\mathbf{J}\Omega\mathbf{r})}{\partial \dot{\mathbf{q}}} = \Lambda(\tau\mathbf{r})\frac{\partial \omega}{\partial \dot{\mathbf{q}}} - \tau\Lambda(\mathbf{r})\frac{\partial \omega}{\partial \dot{\mathbf{q}}}$$

$$= [\Lambda(\tau\mathbf{r}) - \tau\Lambda(\mathbf{r})]\frac{\partial (\tau\nu\sigma_\theta \dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}} = [\Lambda(\tau\mathbf{r}) - \tau\Lambda(\mathbf{r})]\tau\nu\sigma_\theta, \tag{2.153}$$

$$\frac{\partial (\tau^{\mathrm{T}}\mathbf{r})}{\partial \mathbf{q}} = \frac{\partial (\dot{\tau}^{\mathrm{T}}\mathbf{r})}{\partial \dot{\mathbf{q}}} = \frac{\partial (\tau^{\mathrm{T}}\Omega\mathbf{r})}{\partial \dot{\mathbf{q}}} - \frac{\partial (\Omega\tau^{\mathrm{T}}\mathbf{r})}{\partial \dot{\mathbf{q}}}$$

$$= -\tau^T\Lambda(\mathbf{r})\frac{\partial\omega}{\partial\dot{\mathbf{q}}} + \Lambda(-\tau^T\mathbf{r})\frac{\partial\omega}{\partial\dot{\mathbf{q}}} = [-\tau^T\Lambda'_*(\mathbf{r}) + \Lambda(\tau^T\mathbf{r})]\frac{\partial(\tau\nu\sigma_\theta\dot{\mathbf{q}})}{\partial\dot{\mathbf{q}}}$$
$$= [-\tau^T\Lambda(\mathbf{r}) + \Lambda(\tau^T\mathbf{r})]\,\tau\nu\sigma_\theta. \qquad (2.154)$$

The derivative of the vector $\mathbf{A}(\mathbf{q})\,\mathbf{B}(\mathbf{q})\,\mathbf{C}(\mathbf{q})\,\mathbf{R}(\mathbf{q})$ with respect to the parameter $\mathbf{q}$ is

$$\frac{\partial(\mathbf{A}(\mathbf{q})\,\mathbf{B}(\mathbf{q})\,\mathbf{C}(\mathbf{q})\,\mathbf{r}(\mathbf{q}))}{\partial\mathbf{q}} = \frac{\partial^*(\mathbf{A}(\mathbf{q})\,\mathbf{B}(\mathbf{q})\,\mathbf{C}(\mathbf{q})\,\mathbf{r}(\mathbf{q}))}{\partial\mathbf{q}}$$
$$+ \mathbf{A}(\mathbf{q})\frac{\partial^*(\mathbf{B}(\mathbf{q})\,\mathbf{C}(\mathbf{q})\,\mathbf{r}(\mathbf{q}))}{\partial\mathbf{q}} + \mathbf{A}(\mathbf{q})\,\mathbf{B}(\mathbf{q})\frac{\partial^*(\mathbf{C}(\mathbf{q})\,\mathbf{r}(\mathbf{q}))}{\partial\mathbf{q}}$$
$$+ \mathbf{A}(\mathbf{q})\,\mathbf{B}(\mathbf{q})\,\mathbf{C}(\mathbf{q})\frac{\partial\mathbf{r}(\mathbf{q})}{\partial\mathbf{q}}, \qquad (2.155)$$

where the symbol

$$\frac{\partial^*\mathbf{A}(\mathbf{q})\,\mathbf{B}(\mathbf{q})\,\mathbf{C}(\mathbf{q})\,\mathbf{r}(\mathbf{q})}{\partial\mathbf{q}}$$

implies that in all the multipliers of the product being differentiated except the first one the parameter $\mathbf{q}$ is assumed fixed. A similar formula can be obtained for any other number of multipliers. We shall use this expression to calculate the matrices $\mathbf{A}_1$, $\mathbf{B}_1$, $\mathbf{B}_2$, $\mathbf{C}_1$ and $\mathbf{D}$.

Let us find the matrix $\mathbf{A}_1$. Using (2.145), (2.137) and (2.155) we have

$$\mathbf{A}_1 = \frac{\partial[\mathbf{A}(\mathbf{q})\ddot{\mathbf{q}}^*]}{\partial[\partial\mathbf{q}^*\ \mathbf{I}]}\Bigg|_* = \sigma_\theta^T\nu^T\frac{\partial^*(\tau^T\mathbf{J}_0\tau\nu\sigma_\theta\ddot{\mathbf{q}}^*)}{\partial\mathbf{q}}$$
$$+ \sigma_\theta^T\nu^T\tau^T\frac{\partial^*(\mathbf{J}_0\tau\nu\sigma_\theta\ddot{\mathbf{q}}^*)}{\partial\mathbf{q}} + \sigma_\theta^T\nu^T\tau^T\mathbf{J}_0\frac{\partial^*(\tau\nu\sigma_\theta\ddot{\mathbf{q}}^*)}{\partial\mathbf{q}}$$
$$+ \sigma_\theta^T\nu^T\frac{\partial^*(\tau^T(\mathbf{R}+\mathbf{L}\tau^T)\,m\tau\nu\sigma_s\ddot{\mathbf{q}}^*)}{\partial\mathbf{q}} + \sigma_\theta^T\nu^T\tau^T\frac{\partial^*(\mathbf{L}\tau^T m\tau\nu\sigma_s\ddot{\mathbf{q}}^*)}{\partial\mathbf{q}}$$
$$+ \sigma_\theta^T\nu^T\tau^T\mathbf{L}\frac{\partial^*(\tau^T m\tau\nu\sigma_s\ddot{\mathbf{q}}^*)}{\partial\mathbf{q}} + \sigma_\theta^T\nu^T\tau^T(\mathbf{L}\tau^T+\mathbf{R})m\frac{\partial^*(\tau\nu\sigma_s\ddot{\mathbf{q}}^*)}{\partial\mathbf{q}}$$
$$- \sigma_s\nu^T\frac{\partial^*(\tau^T m(\mathbf{R}+\mathbf{L}\tau)\,\tau\nu\sigma_\theta\ddot{\mathbf{q}}^*)}{\partial\mathbf{q}} - \sigma_s\nu^T\tau^T m\frac{\partial^*(\tau\mathbf{L}\tau\nu\sigma_\theta\ddot{\mathbf{q}}^*)}{\partial\mathbf{q}}$$
$$- \sigma_s\nu^T\tau^T m(\mathbf{R}+\tau\mathbf{L})\frac{\partial^*(\tau\nu\sigma_\theta\ddot{\mathbf{q}}^*)}{\partial\mathbf{q}}$$
$$- \sigma_s\nu^T\tau^T m\tau\frac{\partial^*(\mathbf{L}\tau\nu\sigma_\theta\ddot{\mathbf{q}}^*)}{\partial\mathbf{q}} + \sigma_s\nu^T\frac{\partial^*(\tau^T m\tau\nu\sigma_s\ddot{\mathbf{q}}^*)}{\partial\mathbf{q}}$$
$$+ \sigma_s\nu^T\tau^T m\frac{\partial^*(\tau\nu\sigma_s\ddot{\mathbf{q}}^*)}{\partial\mathbf{q}}\Bigg|_*. \qquad (2.156)$$

Observing the equation

$$J_0 = J + (\tau L + R)^T m (\tau L + R) = J - (L\tau^T + R) m (\tau L + R)$$

we find from (2.54), (2.56) and (2.57) for constant J, m and R

$$\begin{aligned}\partial^* (J_0 \tau\nu\sigma_\theta \ddot{q}^*) = \{&\Lambda (\tau^T m (\tau L + R)\tau\nu\sigma_\theta \ddot{q}^*) \nu\sigma_s \\ &- L [\Lambda (\tau^T m (\tau L + R) \tau\nu\sigma_\theta \ddot{q}^*) \\ &- \tau^T \Lambda (m (\tau L + R) \tau\nu\sigma_\theta \ddot{q}^*)] \tau\nu\sigma_\theta \\ &- (L\tau^T + R) m [\Lambda (\tau L\tau\nu\sigma_\theta \ddot{q}^*) \\ &- \tau\Lambda (L\tau\nu\sigma_\theta \ddot{q}^*)] \tau\nu\sigma_\theta \\ &+ (L\tau^T + R) m\tau\Lambda (\tau\nu\sigma_\theta \ddot{q}^*) \nu\sigma_s\}_*. \end{aligned} \tag{2.157}$$

From the expressions (2.156) and (2.157) using (2.151), (2.153) and (2.154) we finely obtain

$$\begin{aligned} A_1 = &\{\sigma_\theta^T \nu^T \{[\Lambda (\tau^T J_0 \tau\nu\sigma_\theta \ddot{q}) - \tau^T \Lambda (J_0 \tau\nu\sigma_\theta \ddot{q})] \tau \\ &- \tau^T L [\Lambda (\tau^T m (\tau L + R) \tau\nu\sigma_\theta \ddot{q}) \\ &- \tau^T \Lambda (m (\tau L + R) \tau\nu\sigma_\theta \ddot{q})] \tau \\ &- \tau^T (L\tau^T + R) m [\Lambda (\tau L\tau\nu\sigma_\theta \ddot{q}) - \tau\Lambda (L\tau\nu\sigma_\theta \ddot{q})] \tau \\ &+ \tau^T J_0 [\tau\Lambda (\nu\sigma_\theta \ddot{q}) - \Lambda (\tau\nu\sigma_\theta \ddot{q}] \tau \\ &+ [\Lambda (\tau^T (L\tau^T + R) m\tau\nu\sigma_s \ddot{q}) \\ &- \tau^T \Lambda ((L\tau^T + R) m\tau\nu\sigma_s \ddot{q})] \tau_i^! \\ &+ \tau^T L [\Lambda (\tau^T m\tau\nu\sigma_s \ddot{q}) - \tau^T \Lambda (m\tau\nu\sigma_s \ddot{q})] \tau \\ &+ \tau^T (L\tau^T + R) m [\Lambda (\tau\nu\sigma_s \ddot{q}) - \tau\Lambda (\nu\sigma_s \ddot{q})] \tau\} |_* \nu\sigma_\theta \\ &+ \sigma^T \nu^T \{\tau^T \Lambda (\tau^T m (\tau L + R) \tau\nu\sigma_\theta \ddot{q}) \\ &+ \tau^T (L\tau^T + R) m\tau\Lambda (\tau\nu\sigma_\theta \ddot{q}) - \tau^T \Lambda (\tau^T m\tau\nu\sigma_s \ddot{q})\} |_* \nu\sigma_{sj}^! \\ &- \sigma_s \nu^T \{[\Lambda (\tau^T m (\tau L + R) \tau\nu\sigma_\theta \ddot{q}) \\ &- \tau^T \Lambda (m (\tau L + R) \tau\nu\sigma_\theta \ddot{q})] \tau \\ &- \tau^T m [\Lambda (\tau L\tau\nu\sigma_\theta \ddot{q}) - \tau\Lambda (L\tau\nu\sigma_\theta \ddot{q})] \tau \\ &- [\Lambda (\tau^T m\tau\nu\sigma_s \ddot{q}) - \tau^T \Lambda (m\tau\nu\sigma_s \ddot{q})] \tau \end{aligned}$$

$$+ \tau^T m \tau L [\Lambda (\tau \nu \sigma_\theta \ddot{q}) - \tau \Lambda (\nu \sigma_\theta \ddot{q})] \tau$$
$$- \tau^T m [\Lambda (\tau \nu \sigma_s \ddot{q}) - \tau \Lambda (\nu \sigma_s \ddot{q})]\} |_* \nu \sigma_\theta$$
$$+ \sigma_s \nu^T [\tau^T m \tau \Lambda (\tau \nu \sigma_\theta \ddot{q})] |_* \nu \sigma_s. \quad (2.158)$$

Substituting from (2.138), (2.145) and (2.155) and denoting for brevity $\tilde{J} = \Omega J - (L\tau^T + R)\, m\, (\Omega R + \tau \Omega L)$ find the matrix

$$B_1 = \frac{\partial B(q, \dot{q})\, \dot{q}^*}{\partial q}\Big|_* = \sigma_\theta \nu^T \frac{\partial^* (\tau^T (J_0 \tau \tilde{\Omega} - \tilde{J})\, \tau \nu \sigma_\theta \dot{q}^*)}{\partial q}$$
$$+ \sigma_\theta \nu^T \tau^T \frac{\partial^* (J_0 \tau \tilde{\Omega} \tau \nu \sigma_\theta \dot{q}^*)}{\partial q} + \sigma_\theta \nu^T \tau^T J_0 \frac{\partial^* (\tau \tilde{\Omega} \tau \nu \sigma_\theta \dot{q}^*)}{\partial q}$$
$$- \sigma_\theta \nu^T \tau^T \frac{\partial^* (\tilde{J} \tau \nu \sigma_\theta \dot{q}^*)}{\partial q} + \sigma_\theta \nu^T \tau^T (J_0 \tau \tilde{\Omega} - \tilde{J}) \frac{\partial^* (\tau \nu \sigma_\theta \dot{q}^*)}{\partial q}$$
$$- 2\sigma_\theta \nu^T \frac{\partial^* (\tau^T (R + L\tau^T)\, m \tau \Omega \nu \sigma_s \dot{q}^*)}{\partial q}$$
$$- 2\sigma_\theta \nu^T \tau^T \frac{\partial^* (L\tau^T m \tau \Omega \nu \sigma_s \dot{q}^*)}{\partial q} - 2\sigma_\theta \nu^T \tau^T L \frac{\partial^* (\tau^T m \tau \Omega \nu \sigma_s \dot{q}^*)}{\partial q}$$
$$- 2\sigma_\theta \nu^T \tau^T (L\tau^T + R)\, m \frac{\partial^* (\tau \Omega \nu \sigma_s \dot{q}^*)}{\partial q}$$
$$- \sigma_s \nu^T \frac{\partial^* (\tau^T m [(R + \tau L)\, \tau \tilde{\Omega} - (\Omega R + \tau \Omega L)]\, \tau \nu \sigma_\theta \dot{q}^*)}{\partial q}$$
$$- \sigma_s \nu^T \tau^T m \left[\frac{\partial^* (\tau L \tau \tilde{\Omega} \tau \nu \sigma_\theta \dot{q}^*)}{\partial q} + \tau \frac{\partial^* (L \tau \tilde{\Omega} \tau \nu \sigma_\theta \dot{q}^*)}{\partial q} \right.$$
$$+ \tau L \frac{\partial^* (\tau \tilde{\Omega} \tau \nu \sigma_\theta \dot{q}^*)}{\partial q} - \frac{\partial^* (\tau \Omega L \tau \nu \sigma_\theta \dot{q}^*)}{\partial q}$$
$$\left. - \tau \Omega \frac{\partial^* (L \tau \nu \sigma_\theta \dot{q}^*)}{\partial q} \right] - \sigma_s \nu^T \tau^T m [(R + \tau L)\, \tau \tilde{\Omega} - (\Omega R + \tau \Omega L)]$$
$$\times \frac{\partial^* (\tau \nu \sigma_\theta \dot{q}^*)}{\partial q} - 2\sigma_s \nu^T \tau^T \frac{\partial^* (\tau^T m \tau \Omega \nu \sigma_s \dot{q}^*)}{\partial q}$$
$$- 2\sigma_s \nu^T \tau^T m \frac{\partial^* (\tau \Omega \nu \sigma_s \dot{q}^*)}{\partial q}\Big|_*. \quad (2.159)$$

Let us find the derivative

$$\frac{\partial^* (J_0 \tau \tilde{\Omega} \tau \nu \sigma_\theta \dot{q}^*)}{\partial q}$$

making use of the expressions (2.151), (2.153) and (2.154) and substituting $\mathbf{J}_0 = \mathbf{J} - (\mathbf{L}\tau^T + \mathbf{R})\,\mathbf{m}\,(\tau\mathbf{L} + \mathbf{R})$

$$\frac{\partial^*(\mathbf{J}_0\tau\tilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)}{\partial\mathbf{q}}$$

$$= \Lambda\,(\tau^T\mathbf{m}\,(\tau\mathbf{L}+\mathbf{R})\,\tau\tilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)\,\nu\sigma_s$$

$$- \mathbf{L}[\Lambda'_s(\tau^T\mathbf{m}\,(\tau\mathbf{L}+\mathbf{R})\,\tau\tilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)\,\nu\sigma_s - \tau^T\Lambda(\mathbf{m}(\tau\mathbf{L}+\mathbf{R})\tau\tilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)]\tau\nu\sigma_\theta$$

$$- (\mathbf{L}\tau^T + \mathbf{R})\,\mathbf{m}\,[\Lambda\,(\tau\mathbf{L}\tau\tilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}}^*) - \tau\Lambda\,(\tau\mathbf{L}\tau\tilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)]\,\tau\nu\sigma_\theta$$

$$+ (\mathbf{L}\tau^T + \mathbf{R})\,\mathbf{m}\tau\Lambda\,(\tau\tilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)\,\nu\sigma_s. \qquad (2.160)$$

Considering that $\tilde{\mathbf{J}} = \Omega\mathbf{J} - (\mathbf{L}\tau^T + \mathbf{R})\,\mathbf{m}\,(\Omega\mathbf{R} + \tau\Omega\mathbf{L})$ and making use of (2.149), (2.150), and (2.153-155) calculate the derivative

$$\frac{\partial^*(\tilde{\mathbf{J}}\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)}{\partial\mathbf{q}}$$

$$= -\frac{\partial^*(\mathbf{L}\tau^T\mathbf{m}\,(\Omega\mathbf{R}+\tau\Omega\mathbf{L})\,\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)}{\partial\mathbf{q}} - \mathbf{L}\,\frac{\partial^*(\tau^T\mathbf{m}\,(\Omega\mathbf{R}+\tau\Omega\mathbf{L})\,\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)}{\partial\mathbf{q}}$$

$$- (\mathbf{R} + \mathbf{L}\tau^T)\,\mathbf{m}\left[\frac{\partial^*(\tau\Omega\mathbf{L}\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)}{\partial\mathbf{q}} + \tau\Omega\,\frac{\partial^*(\mathbf{L}\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)}{\partial\mathbf{q}}\right]$$

$$= \Lambda\,(\tau^T\mathbf{m}'_s(\Omega\mathbf{R} + \tau\Omega\mathbf{L})\,\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)\,\nu\sigma_s - \mathbf{L}\,[\Lambda\,(\tau^T\mathbf{m}\,(\Omega\mathbf{R} + \tau\Omega\mathbf{L})\,\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)$$

$$- \tau^T\Lambda\,(\mathbf{m}\,(\Omega\mathbf{R} + \tau\Omega\mathbf{L})\,\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)]\,\tau\nu\sigma_\theta - (\mathbf{R} + \tau^T\mathbf{L})\,\mathbf{m}\,(\Lambda\,(\tau\Omega\mathbf{L}\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)$$

$$- \tau\Lambda\,(\Omega\mathbf{L}\tau\nu\sigma_\theta\dot{\mathbf{q}}^*))\,\tau\nu\sigma_\theta - \tau\Omega\Lambda\,(\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)\,\nu\sigma_s. \qquad (2.161)$$

From (2.159), (2.160) and (2.161) observing the expressions for the derivatives in (2.150-154) we find the expression to compute the matrix

$$\mathbf{B}_1 = \sigma_\theta\nu^T\{[\Lambda'_s(\tau^T\,(\mathbf{J}_0\tau\tilde{\Omega} - \tilde{\mathbf{J}})\,\tau\nu\sigma_\theta\dot{\mathbf{q}})$$

$$- \tau^T\Lambda\,((\mathbf{J}_0\tau\tilde{\Omega} - \tilde{\mathbf{J}})\,\tau\nu\sigma_\theta\dot{\mathbf{q}})]\,\tau + \tau^T\,[\tau^T\Lambda\,(\mathbf{m}\,(\tau\mathbf{L} + \mathbf{R})$$

$$\times\,\tau\tilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}}) - \Lambda\,(\tau^T\Lambda\,(\mathbf{m}\,(\tau\mathbf{L} + \mathbf{R})\,\tau\tilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}}))]\,\tau$$

$$- (\mathbf{L}\tau^T + \mathbf{R})\,\mathbf{m}\,[\Lambda\,(\tau\mathbf{L}\tau\tilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}}) - \tau\Lambda\,(\tau\mathbf{L}\tau\tilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}})]\,\tau$$

$$+ \tau^T\mathbf{J}_0\,[\Lambda\,(\tau\tilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}}) - \tau\Lambda\,(\tilde{\Omega}\tau\nu\sigma_\theta\mathbf{q})\,\dot{\tau}$$

$$- \tau^T\,[\tau^T\Lambda\,(\mathbf{m}\,(\Omega\mathbf{R} + \tau\Omega\mathbf{L})\,\tau\nu\sigma_\theta\dot{\mathbf{q}}) - \Lambda\,(\tau\mathbf{m}\,(\Omega\mathbf{R} + \tau\Omega\mathbf{L})\,\tau\nu\dot{\sigma}_\theta\dot{\mathbf{q}})\,\tau$$

$$- \tau^T\,(\mathbf{L}\tau^T + \mathbf{R})\,\mathbf{m}\,[-\Lambda\,(\tau\Omega\mathbf{L}\tau\nu\sigma_\theta\dot{\mathbf{q}}) + \tau\Lambda\,(\Omega\mathbf{L}\tau\nu\sigma_\theta\dot{\mathbf{q}})]\,\tau$$

$$+ \tau^T\,(\mathbf{J}_0\tau\Omega - \tilde{\mathbf{J}})\,[\Lambda'_s(\tau\nu\sigma_\theta\dot{\mathbf{q}}) - \tau\Lambda\,(\nu\sigma_\theta\dot{\mathbf{q}})]\,\tau$$

$$
\begin{aligned}
&-\tau^{\mathrm{T}}\mathbf{J}_0\tau\Lambda\,(\tau\nu\sigma_\theta\dot{\mathbf{q}})-2\,[\Lambda\,(\tau^{\mathrm{T}}\,(\mathbf{R}+\mathbf{L}\tau^{\mathrm{T}})\,\mathbf{m}\tau\Omega\nu\sigma_s\dot{\mathbf{q}})\\
&-\tau^{\mathrm{T}}\Lambda\,((\mathbf{R}+\mathbf{L}\tau^{\mathrm{T}})\,\mathbf{m}\tau\Omega\nu\sigma_s\dot{\mathbf{q}})]\,\tau-2\tau^{\mathrm{T}}\nu\tau^{\mathrm{T}}\mathbf{L}\,[\Lambda\,(\tau^{\mathrm{T}}\mathbf{m}\tau\Omega\nu\sigma_s\dot{\mathbf{q}})\\
&-\tau^{\mathrm{T}}\Lambda\,(\mathbf{m}\tau\Omega\nu\sigma_s\dot{\mathbf{q}})]\,\tau-2\tau^{\mathrm{T}}\,(\mathbf{L}\tau^{\mathrm{T}}+\mathbf{R})\,\mathbf{m}\,[\Lambda\,(\tau\Omega\nu\sigma_s\dot{\mathbf{q}})\\
&-\Lambda\,(\Omega\nu\sigma_s\dot{\mathbf{q}})]\,\tau\}_*\,\nu\sigma_\theta+\sigma_\theta\nu^{\mathrm{T}}\,\{\tau^{\mathrm{T}}\Lambda\,(\tau^{\mathrm{T}}\mathbf{m}\,(\tau\mathbf{L}+\mathbf{R})\,\tau\widetilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}})\\
&+(\mathbf{L}\tau^{\mathrm{T}}+\mathbf{R})\,\mathbf{m}\tau\Lambda\,(\tau\Omega\tau\nu\sigma_\theta\dot{\mathbf{q}})-\tau^{\mathrm{T}}\Lambda\,(\tau^{\mathrm{T}}\mathbf{m}\,(\Omega\mathbf{R}+\tau\Omega\mathbf{L})\,\tau\nu\sigma_\theta\dot{\mathbf{q}})\\
&-\tau^{\mathrm{T}}\,(\mathbf{L}\tau^{\mathrm{T}}+\mathbf{R})\,\mathbf{m}\tau\Omega\Lambda\,(\tau\nu\sigma_\theta\dot{\mathbf{q}})+2\tau^{\mathrm{T}}\sigma_\theta\nu^{\mathrm{T}}\tau^{\mathrm{T}}\Lambda\,(\tau^{\mathrm{T}}\mathbf{m}\tau\Omega\nu\sigma_s\dot{\mathbf{q}})\}_*\,\nu\sigma_s\\
&-\sigma_s^{\mathrm{T}}\nu^{\mathrm{T}}\,\{[\Lambda\,(\tau^{\mathrm{T}}\mathbf{m}\,[(\mathbf{R}+\tau\mathbf{L})\,\tau\widetilde{\Omega}-(\Omega\mathbf{R}+\tau\Omega\mathbf{L})]\,\tau\nu\sigma_\theta\dot{\mathbf{q}})\\
&-\tau^{\mathrm{T}}\Lambda\,(\mathbf{m}\,[(\mathbf{R}+\tau\mathbf{L})\,\tau\widetilde{\Omega}-(\Omega\mathbf{R}+\Omega\tau\mathbf{L})]\,\tau\nu\sigma_\theta\dot{\mathbf{q}})\,\tau\\
&+\tau^{\mathrm{T}}\mathbf{m}\,[\Lambda\,(\tau\mathbf{L}\tau\widetilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}})-\tau\Lambda\,(\mathbf{L}\tau\widetilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}})]\,\tau-\tau^{\mathrm{T}}\mathbf{m}\tau\,[\Lambda\,(\tau\widetilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}})\\
&-\tau\Lambda\,(\widetilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}})]\,\tau+\tau^{\mathrm{T}}\mathbf{m}\,[\Lambda(\tau\Omega\mathbf{L}\tau\nu\sigma_\theta\dot{\mathbf{q}})-\tau\Lambda\,(\Omega\mathbf{L}\tau\nu\sigma_\theta\dot{\mathbf{q}})]\,\tau\\
&-2\tau^{\mathrm{T}}\mathbf{m}\,[\Lambda\,(\tau\Omega\nu\sigma_s\dot{\mathbf{q}})-\tau\Lambda\,(\Omega\nu\sigma_s\dot{\mathbf{q}})]\,\tau\\
&-\tau^{\mathrm{T}}\mathbf{m}\,[(\mathbf{R}+\tau\mathbf{L})\,\tau\widetilde{\Omega}-(\Omega\mathbf{R}+\tau\Omega\mathbf{L})]\,[\Lambda\,(\tau\Omega\nu\sigma_s\dot{\mathbf{q}})-\tau\Lambda\,(\Omega\nu\sigma_s\dot{\mathbf{q}})]\,\tau\\
&-2\tau^{\mathrm{T}}\,[\Lambda\,(\tau^{\mathrm{T}}\mathbf{m}\tau\Omega\nu\sigma_s\dot{\mathbf{q}})-\tau^{\mathrm{T}}\Lambda\,(\mathbf{m}\tau\Omega\nu\sigma_s\dot{\mathbf{q}})]\,\tau\}_*\,\nu\sigma_\theta\\
&+\sigma_s^{\mathrm{T}}\nu^{\mathrm{T}}\,\{\tau^{\mathrm{T}}\mathbf{m}\tau\Lambda\,(\widetilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}})+\tau^{\mathrm{T}}\mathbf{m}\tau\Omega\Lambda\,(\tau\nu\sigma_\theta\dot{\mathbf{q}})\}_*\,\nu\sigma_s. \qquad (2.162)
\end{aligned}
$$

Now we find the matrix $\mathbf{B}_2$ from (2.138) and (2.145) in the form

$$
\begin{aligned}
\mathbf{B}_2=\frac{\partial\,(\mathbf{B}\,(\mathbf{q},\,\dot{\mathbf{q}})\,\dot{\mathbf{q}}^*)}{\partial\dot{\mathbf{q}}}\Bigg|&=\sigma_\theta\nu^{\mathrm{T}}\tau^{\mathrm{T}}\Big\{\mathbf{J}_0\tau\,\frac{\partial^*\,(\widetilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)}{\partial\dot{\mathbf{q}}}\\
&-\frac{\partial^*\,(\Omega\mathbf{J}\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)}{\partial\dot{\mathbf{q}}}-(\tau\mathbf{L}+\mathbf{R})^{\mathrm{T}}\,\mathbf{m}\,\frac{\partial^*\,(\Omega\mathbf{R}\tau\nu\sigma_\theta\dot{\mathbf{q}})}{\partial\dot{\mathbf{q}}}\\
&-(\tau\mathbf{L}+\mathbf{R})^{\mathrm{T}}\,\mathbf{m}\tau\,\frac{\partial^*\,(\Omega\mathbf{L}\tau\nu\sigma_\theta\dot{\mathbf{q}})}{\partial\dot{\mathbf{q}}}\Big\}_*+2\sigma_\theta\nu^{\mathrm{T}}\tau^{\mathrm{T}}\,(\mathbf{R}+\tau\mathbf{L})^{\mathrm{T}}\,\mathbf{m}\tau\\
&\times\frac{\partial^*\,(\Omega\nu\sigma_s\dot{\mathbf{q}}^*)}{\partial\dot{\mathbf{q}}}\Bigg|_*-\sigma_s\nu^{\mathrm{T}}\tau^{\mathrm{T}}\mathbf{m}\,(\mathbf{R}+\tau\mathbf{L})\,\tau\\
&\times\frac{\partial^*\,(\widetilde{\Omega}\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)}{\partial\dot{\mathbf{q}}}\Bigg|_*+\sigma_s\nu^{\mathrm{T}}\tau^{\mathrm{T}}\,\mathbf{m}\,\frac{\partial^*\,(\Omega\mathbf{L}\tau\nu\sigma_\theta\dot{\mathbf{q}}^*)}{\partial\dot{\mathbf{q}}}\Bigg|_*\\
&-2\sigma_s\nu^{\mathrm{T}}\tau^{\mathrm{T}}\mathbf{m}\tau\,\frac{\partial^*\,(\Omega\nu\sigma_s\dot{\mathbf{q}}^*)}{\partial\dot{\mathbf{q}}}\Bigg|_*
\end{aligned}
\qquad (2.163)
$$

Then substituting from (2.148), (2.149) and (2.163) we finely get

$$\begin{aligned}\mathbf{B}_2 = \sigma_\theta^T \nu^T \tau_*^T \{ &- \mathbf{J}_0 \tau \Lambda (\tau \nu \sigma_\theta \dot{\mathbf{q}}^*) + \Lambda (\mathbf{J} \tau \nu \sigma_\theta \dot{\mathbf{q}}) \tau \\ &+ \Lambda (\mathbf{J} \tau \nu \sigma_\theta \dot{\mathbf{q}}) \tau + (\tau \mathbf{L} + \mathbf{R})^T \mathbf{m} \Lambda (\mathbf{R} \tau \sigma_\theta \dot{\mathbf{q}}) \tau \\ &+ (\tau \mathbf{L} + \mathbf{R})^T \mathbf{m} \tau \Lambda (\mathbf{L} \tau \nu \sigma_\theta \dot{\mathbf{q}}) - 2 (\tau \mathbf{L} + \mathbf{R})^T \mathbf{m} \tau \Lambda (\nu \sigma_s \dot{\mathbf{q}}) \tau \}_* \nu \sigma_\theta \\ &+ \sigma_s \nu^T \tau_*^T \{ \mathbf{m} (\mathbf{R} + \tau \mathbf{L}) \tau \Lambda (\tau \nu \sigma_\theta \dot{\mathbf{q}}) - \mathbf{m} \Lambda (\mathbf{R} \tau \nu \sigma_\theta \dot{\mathbf{q}}) \tau \\ &- \mathbf{m} \tau \Lambda (\mathbf{L} \tau \nu \sigma_\theta \dot{\mathbf{q}}) \tau + 2 \mathbf{m} \tau \Lambda (\nu \sigma_s \dot{\mathbf{q}}) \tau \}_* \nu \sigma_\theta. \end{aligned} \tag{2.164}$$

From the equations (2.139), (2.140) and (2.145), (2.146) and (2.155) we calculate the matrices $\mathbf{C}_1$ and $\mathbf{D}_1$.

$$\mathbf{C}_1 = \frac{\partial [\mathbf{C}(\mathbf{q}) \mathbf{M}_e^0]}{\partial \mathbf{q}} = \sigma_\theta \nu^T \frac{\partial (\tau_0 \mathbf{Z}^T \mathbf{M}_e^0)}{\partial \mathbf{q}},$$

or taking into account that $\mathbf{Z}$ = constant and observing (2.152) we get

$$\mathbf{C}_1 = \sigma_\theta \nu^T \Lambda (\tau_0 \mathbf{Z}^T \mathbf{M}_e^{0*}) \tau \nu \sigma_\theta \,|_*. \tag{2.165}$$

$$\begin{aligned}\mathbf{D}_1 = \frac{\partial [\mathbf{D}(\mathbf{q}) (\mathbf{F}_e^{0*} + \mathbf{G}^{0*})]}{\partial \mathbf{q}}\Big|_* &= \Big\{ \sigma_s \nu^T \frac{\partial^* [\tau_0 \mathbf{Z}^T (\mathbf{F}_e^{0*} + \mathbf{G}^{0*})]}{\partial \mathbf{q}} \\ &- \sigma_\theta \nu^T \frac{\partial^* [\tau^T (\mathbf{R} + \tau \mathbf{L})^T \tau_0 (\mathbf{F}_e^{0*} + \mathbf{G}^{0*})]}{\partial \mathbf{q}} \\ &+ \sigma_\theta \nu^T \tau^T \frac{\partial^* [\mathbf{L} \tau^T \tau_0 (\mathbf{F}_e^{0*} + \mathbf{G}^{0*})]}{\partial \mathbf{q}} + \sigma_\theta \nu^T \tau^T \mathbf{L} \frac{\partial^* [\tau^T \tau_0 (\mathbf{F}_e^{0*} + \mathbf{G}^{0*})]}{\partial \mathbf{q}} \Big\}_* \\ = \Big\{ \sigma_s \nu^T &\frac{\partial^* [\tau_0 \mathbf{Z}^T (\mathbf{F}_e^{0*} + \mathbf{G}^{0*})]}{\partial \mathbf{q}} \\ &+ \sigma_\theta \nu^T \tau^T \frac{\partial^* [\mathbf{L} \tau_0 \mathbf{Z}^T (\mathbf{F}_e^{0*} + \mathbf{G}^{0*})]}{\partial \mathbf{q}} + \sigma_\theta \nu^T \mathbf{L} \frac{[\tau_0 \mathbf{Z}^T (\mathbf{F}_e^{0*} + \mathbf{G}^{0*})]}{\partial \mathbf{q}} \\ &+ \sigma_\theta \nu^T \frac{\partial^* [\tau^T (\mathbf{L} \tau^T + \mathbf{R}) \tau_0 (\mathbf{F}_e^{0*} + \mathbf{G}^{0*})]}{\partial \mathbf{q}} \Big\}_*. \end{aligned}$$

Substituting from (2.150-152) yields

$$\begin{aligned}\mathbf{D}_1 = \sigma_s \nu^T \{ &\Lambda (\tau_0 \mathbf{Z}^T (\mathbf{F}_e^0 + \mathbf{G}^0)) \}_* \nu \sigma_\theta - \sigma_\theta \nu^T \{ \tau^T \Lambda (\tau_0 \mathbf{Z}^T (\mathbf{F}_e^0 + \mathbf{G}^0)) \} \nu \sigma_s \\ &- \sigma_\theta \nu^T \{ \tau^T \Lambda (\tau_0 \mathbf{Z}^T (\mathbf{F}_e^0 + \mathbf{G}^0)) - [\Lambda (\tau^T (\mathbf{L} \tau^T + \mathbf{R}) \tau_0 (\mathbf{F}_e^0 + \mathbf{G}^0)) \\ &- \tau^T \Lambda ((\mathbf{L} \tau^T + \mathbf{R}) \tau_0 (\mathbf{F}_e^0 + \mathbf{G}^0))] \}_* \tau \nu \sigma_\theta. \end{aligned} \tag{2.166}$$

In the particular case of linearization about an equilibrium state, that is with $\mathbf{q}^*$ = constant, and $\dot{\mathbf{q}}^* = \ddot{\mathbf{q}}^* = 0$ and the sole forces of gravity acting on the mechanism ($\mathbf{M}_e^{0*} = \mathbf{F}_e = 0$), the equations

(2.158), (2.162) and (2.164-166) become

$$A_1 = 0, \quad B_1 = 0, \quad B_2 = 0, \quad C_1 = 0,$$

$$\begin{aligned} D_1 = {} & \sigma_\theta \nu^T \{[\Lambda (\tau^T (L\tau^T + R) \tau_0 G^0) - \tau^T \Lambda (L\tau^T + R) \tau_0 G^0 \\ & - \tau^T L \Lambda (\tau_0 Z^T G^0)\}_* \tau_* \nu \sigma_\theta + \sigma_s \nu^T \Lambda (\tau_0 Z^T G^0) \tau \nu \sigma_\theta |_* \\ & - \sigma_\theta \nu^T \tau^T \Lambda (\tau_0 Z^T G^0) \nu \sigma_s |_*. \end{aligned} \tag{2.167}$$

The coefficients of the linearized differential equation become in this case

$$a(q^*) = A^*, \quad b(q^*, 0) = -B^*, \quad c(q^*, 0, 0) = -D_1. \tag{2.168}$$

The derived expressions for the coefficients of the differential equation (2.146) can be used to compute these coefficients on a computer.

The developed mathematical relationships are a valuable tool to investigate the dynamical behavior of robotic manipulator used to select the characteristics of arm actuators, to devise the control strategies, and to develop microprocessor programs.

List of Symbols

q	vector of generalized coordinates
v	vector of linear velocity
w	vector of linear acceleration
ω	vector of angular velocity
ε	vector of angular acceleration
τ	matrix of transformation from one reference frame to another
J	matrix of inertia
l	vector of link lengths
ρ	vector of mass centers
μ	vector of forces and moments in joints
0	superscript "0" refers to the inertial reference frame

3 A Frequency-domain Approach to Synthesis of Robot Actuator System

V. S. Medvedev, *D. Sc. (Tech.)*, *and*
L. P. Shigorin, *Cand. Sc. (Tech.)*

3.1 GENERAL

This chapter is devoted to the synthesis of control systems for robot arm actuators. Our objective is to provide for the system a specified performance and stability when the parameters of the arm linkage vary over wide range.

A popular approach to design of cross-coupled multivariable systems involves analysis in the frequency domain (FD), in which interactions of separate control systems are evaluated plotting their log frequency characteristics [3]. This work proposes to investigate stability of a multivariable system applying the modified Nyquist criterion to the characteristic matrix.

A setback of the method, however, is in that the computer aided (CAD) synthesis of correcting means is difficult to realize since the designer has to analyze n^2 intercoupled channels of the system with n separate (one input—one output) elements and $n\,(n-1)$ cross-couplings.

On the other hand, a time-domain approach to the synthesis of correcting means for high order multivariable systems bumps into severe difficulties because of high dimension of the problem: the large size matrices result in time consuming computing and high load on computer memory.

The above argument indicates the necessity to develop new design methods for robot actuators, which, on the one side, enable the designer to operate relatively low dimension matrices and, on the other, make a platform to build up efficient CAD procedures of correcting means synthesis.

In the following sections we shall consider a method designed to choose the parameters of robot actuators with the help of the spectral frequency responses of multivariable system and present a generalization of the method of natural modes Ω_i and damping factors ζ_i utilized to design multivariable mechanical systems in frequency domain [51]. The generalization became possible owing to that the actuator system had its differential equation model reduced to the form similar to that describing purely mechanical systems.

Consider a model of robot arm which being linearized at an equilibrium point (or about some reference path) assumes the form of

a vector differential equation (see Chap. 2)

$$\mathbf{a}(\mathbf{q}^*)\ddot{\mathbf{q}}(t)+\mathbf{b}(\mathbf{q}^*,\dot{\mathbf{q}}^*)\dot{\mathbf{q}}(t)+\mathbf{c}(\mathbf{q}^*,\dot{\mathbf{q}}^*,\ddot{\mathbf{q}}^*)\mathbf{q}(t)$$
$$=\mathbf{W}_M(D)[\mathbf{g}(t)-\mathbf{q}(t)]-\mathbf{W}_F(D)\mathbf{q}(t)+\mathbf{M}_e(t), \quad (3.1)$$

where $\mathbf{q}^*$, $\dot{\mathbf{q}}^*$, and $\ddot{\mathbf{q}}^*$ are the vectors of the system generalized coordinates and their derivatives, corresponding to the reference path of manipulator motion; $D = d/dt$ is the differential operator; $\mathbf{a}(\mathbf{q}^*)$, $\mathbf{b}(\mathbf{q}^*, \dot{\mathbf{q}}^*)$, and $\mathbf{c}(\mathbf{q}^*, \dot{\mathbf{q}}^*, \ddot{\mathbf{q}}^*)$ are the matrices describing the dynamic behavior of the robot arm; $\mathbf{W}_M(D)$ and $\mathbf{W}_F(D)$ are the transfer matrices describing the dynamics which the forward paths and common intraneous loop of the open loop actuators (without correction) display towards the arm linkage as the load; $\mathbf{g}(t)$ is the vector of controls; $\mathbf{q}(t)$ is the vector arraying the deviations of the actuator generalized coordinates from the reference values, $\mathbf{M}_e$ is the vector of external moments. Henceforth we will assume that the number of the generalized coordinates amounts to n.

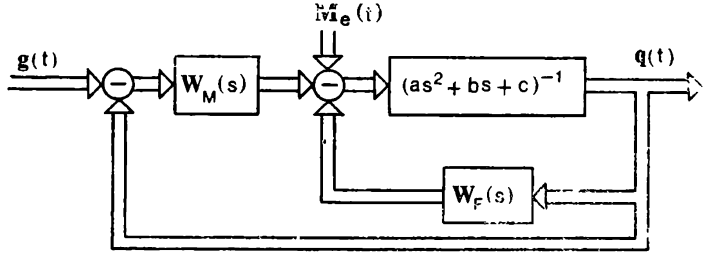


Fig. 3.1 Block diagram of robot's actuator system without correction

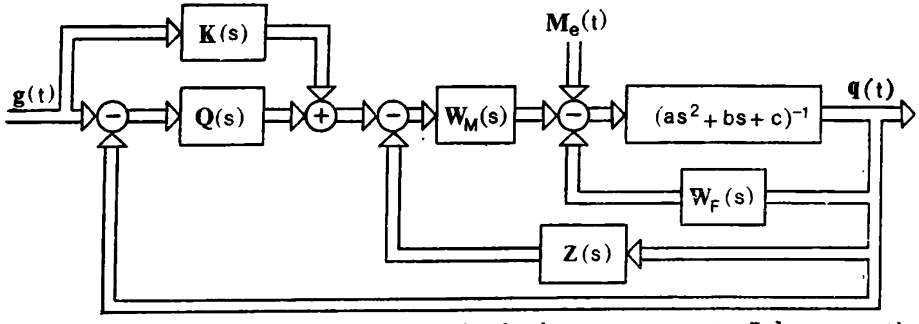


Fig. 3.2 Block diagram of robot's actuator system plus correction

The block diagram of the system mechanism, described by the equation (3.1), is shown in Fig. 3.1, and that of the actuators in Fig. 3.2. Here, $\mathbf{Q}(D)$ is the matrix of functions for sequential and $\mathbf{Z}(D)$ that for parallel correction, $\mathbf{K}(D)$ is the matrix of functions for the correction with respect to the controls $\mathbf{g}(t)$.

A vector differential equation for the system of Fig. 3.2 has the form

$$[\mathbf{a}(\mathbf{q}^*)\,D^2 + \mathbf{b}(\mathbf{q}^*, \dot{\mathbf{q}}^*)\,D + \mathbf{c}(\mathbf{q}^*, \dot{\mathbf{q}}^*, \ddot{\mathbf{q}}^*)]\,\mathbf{q}(t)$$
$$= -[\mathbf{W}_F(D) + \mathbf{W}_M(D)\,\mathbf{Z}(D)]\,\mathbf{q}(t) + \mathbf{W}_M(D)\,\mathbf{Q}(D)\,[\mathbf{g}(t) - \mathbf{q}(t)]$$
$$+ \mathbf{W}_M(D)\,\mathbf{K}(D)\,\mathbf{g}(t) + \mathbf{M}_e(t). \qquad (3.2)$$

The synthesis in this formulation consists in finding the elements of the matrices **Q** (D), **Z** (D) and **K** (D) that generate the specified performance for the robot actuator system.

The method of spectral frequency responses (to be defined below) treated in this chapter is based on an analogy between the description of a purely mechanical multivariable system by the model

$$(\mathbf{a}D^2 + \mathbf{b}D + \mathbf{c})\,\mathbf{q}(t) = \boldsymbol{\mu}(t),$$

where **a**, **b**, and **c** are the constant matrices of the system inertial, damping, and flexibility characteristics, and the model of the multivariable actuator system of a robot (3.2), which reduces here to the form

$$[\mathbf{a}D^2 + \mathbf{B}(D)\,D + \mathbf{C}(D)]\,\mathbf{q}(t) = \boldsymbol{\mu}_1(t).$$

Here, the operator matrices **B** (D) and **C** (D) of the system viscosities and flexibilities are no longer constant as it was the case with the purely mechanical system, rather, they depend on the parameter D. In the Laplace transform domain these matrices will contain real functions of angular frequency ω. It implies that for a given ω the matrices **a**, **B** (ω) and **C** (ω) will describe that purely mechanical system whose response to a harmonic stimulus at the frequency ω is identical to the behavior of the actuator system under study at this frequency.

As a consequence, the spectral characteristics of the multivariable actuator system, that is, its natural modes Ω_i and damping factors ζ_i, $i = 1, 2, \ldots, n$, obtained, as with the mechanical system, by transforming the matrix of homogeneous differential equations to a diagonal form [51], also depend upon frequency, i.e.

$$\Omega_i = \Omega_i(\omega), \quad \zeta_i = \zeta_i(\omega), \quad i = 1, 2, \ldots, n.$$

The thus obtained functions $\Omega_i(\omega)$ and $\zeta_i(\omega)$ which we shall refer to in the following as *spectral frequency responses* (SFR), define, as will be shown below, the stability of the system and provide for the designer a tool to analyze the system behavior.

To demonstrate, at $\omega = 0$ these characteristics yield the dynamics of the purely mechanical system. In the frequency range from $\omega = 0$ to that where the responses $\Omega_i(\omega)$ firstly intersect the line $\Omega_i = \omega$ they describe the behavior of the system involving the multivariable purely mechanical load (arm linkage) and the actuators with the

intrinsic dynamics. The subsequent (as ω increases) crossings of $\Omega_i(\omega)$ with $\Omega_i = \omega$ supply an information on the behavior of the system influenced by the dynamics of actuator-plus-controller systems in the high frequency range.

The spectral frequency responses give also a clue to the synthesis of the correction laws which improve the performance of the entire robot actuator system. The procedure of the synthesis will also be covered in this chapter.

In the following we shall give a more rigorous treatment to the method of spectral frequency responses.

3.2 SPECTRAL FREQUENCY RESPONSE ANALYSIS

Consider the linearized equations (3.2) describing the dynamics of robot controller-plus-actuator system. Having denoted

$$\mathbf{W_M}(D)\,[\mathbf{Q}(D) + \mathbf{K}(D)] = \mathbf{G}(D);$$

$$\mathbf{W_F}(D) + \mathbf{W_M}(D)\,\mathbf{Z}(D) + \mathbf{W_M}(D)\,\mathbf{Q}(D) = \mathbf{W_\Sigma}(D), \tag{3.3}$$

the vector equation (3.2) reduces to

$$[\mathbf{a}(\mathbf{q}^*)\,D^2 + \mathbf{b}(\mathbf{q}^*, \dot{\mathbf{q}}^*)\,D + \mathbf{c}(\mathbf{q}^*, \dot{\mathbf{q}}^*, \ddot{\mathbf{q}}^*) + \mathbf{W_\Sigma}(D)]\,\mathbf{q}(t) = \mathbf{G}(D)\,\mathbf{g}(t) + \mathbf{M_e}(t). \tag{3.4}$$

Notice that when the correcting facilities are absent, Eqs. (3.3) simplify since $\mathbf{Q}(D) = \mathbf{I}$, $\mathbf{Z}(D) = \mathbf{0}$, and $\mathbf{K}(D) = \mathbf{0}$; here $\mathbf{I}$ is the identity matrix. Notice also that the elements in the matrices $\mathbf{W_F}(D)$, $\mathbf{W_M}(D)$, $\mathbf{Z}(D)$, $\mathbf{Q}(D)$, $\mathbf{K}(D)$, $\mathbf{G}(D)$, and $\mathbf{W_\Sigma}(D)$ generally are rational functions.

Representing the transfer operator matrix $\mathbf{W_\Sigma}(D)$ as

$$\mathbf{W_\Sigma}(D) = \mathbf{W_1}(D)\,D + \mathbf{W_2}(D) \tag{3.5}$$

and denoting

$$\mathbf{A} = \mathbf{a}(\mathbf{q}^*)$$

$$\mathbf{B}(D) = \mathbf{W_1}(D) + \mathbf{b}(\mathbf{q}^*, \dot{\mathbf{q}}^*) \tag{3.6}$$

$$\mathbf{C}(D) = \mathbf{W_2}(D) + \mathbf{c}(\mathbf{q}^*, \dot{\mathbf{q}}^*, \ddot{\mathbf{q}}^*)$$

the equation (3.4) may be rewritten as

$$[\mathbf{A}D^2 + \mathbf{B}(D)\,D + \mathbf{C}(D)]\,\mathbf{q}(t) = \mathbf{G}(D)\,\mathbf{g}(t) + \mathbf{M_e}(t). \tag{3.7}$$

In a diagrammatic form this equation is shown in Fig. 3.3. It describes the set of robot arm actuators as a multivariable system of n

degrees of freedom. The matrices **B** (D) and **C** (D), whose elements depend on the differential operator D, shall be referred to as dynamic viscosity and dynamic flexibility matrices.

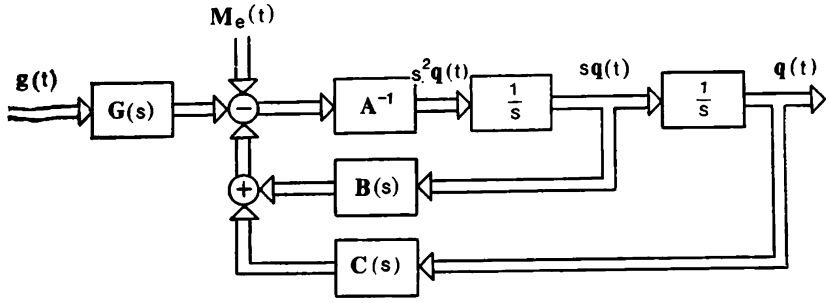


Fig. 3.3 Block diagram of a structure shown in Fig. 3.2

Setting in (3.7) $\mathbf{W}_1(D) = \mathbf{W}_2(D) = \mathbf{0}$ [see the convention (3.6)] and denoting

$$\boldsymbol{\mu}(t) = \mathbf{G}(D)\,\mathbf{g}(t) + \mathbf{M_e}(t), \tag{3.8}$$

yields the vector equation

$$(\mathbf{a}D^2 + \mathbf{b}D + \mathbf{c})\,\mathbf{q}(t) = \boldsymbol{\mu}(t), \tag{3.9}$$

which describes the dynamics of robot actuator system with n degrees of freedom displayed under the action of $\boldsymbol{\mu}(t)$, moments about separate kinematic pairs. The behavior of this system can be evaluated by $2n$ roots of the characteristic equation

$$\det(\mathbf{a}\lambda^2 + \mathbf{b}\lambda + \mathbf{c}) = 0, \tag{3.10}$$

which also define n natural modes Ω_i and n damping factors ζ_i ($i = 1, 2, \ldots, n$).

When the moments about the joints are generated by the actuators, that is $\mathbf{W}_1(D)$ and $\mathbf{W}_2(D)$ are nonsingular the dynamic behavior of the whole control system may be described by the respective characteristics $\Omega_i(D)$ and $\zeta_i(D)$ obtained from (3.10) which now, in contrast to the case of purely mechanical load, will be functions of operator D. We will refer to these parameters as the spectral dynamic response of the system modelled by (3.9).

Consider now the stability of the system described by the vector linear differential equation (3.9). We will assume the external disturbances to be absent, i.e. $\mathbf{M_e} = \mathbf{0}$.

Let $V(D)$ be the least common multiple of the denominators in the elements of matrices **B** (D) and **C** (D), and $G_2(D)$ be that of the elements of matrix **G** (D). Both $V(D)$ and $G_2(D)$ are, obviously,

scalar polynomials in D. With the introduced notation we can write

$$\mathbf{B}(D)=\frac{\mathbf{B}_1(D)}{V(D)},\quad \mathbf{C}(D)=\frac{\mathbf{C}_1(D)}{V(D)},\quad \mathbf{G}(D)=\frac{\mathbf{G}_1(D)}{G_2(D)}, \tag{3.11}$$

where the elements of matrices $\mathbf{B}_1(D)$, $\mathbf{C}_1(D)$ and $\mathbf{G}_1(D)$ are polynomials in D. The equation (3.8) may be rewritten, too

$$\mathbf{G}_2(D)\,[V(D)\,\mathbf{A}D^2+\mathbf{B}_1(D)\,D+\mathbf{C}_1(D)]\,\mathbf{q}(t) = V(D)\,\mathbf{G}_1(D)\,\mathbf{g}(t). \tag{3.12}$$

Substituting the Laplace variable $s=\alpha+j\omega$ for D into (3.12) we assert that for the system described by this equation to be stable it is necessary and sufficient that all the roots of the polynomials $G_2(s)$ and $\det[V(s)\,\mathbf{A}s^2+\mathbf{B}_1(s)\,s+\mathbf{C}_1(s)]$ should be in the left half plane of the complex s variable.

Suppose that all the roots of polynomial $G_2(s)$ satisfy this condition, then to find out whether or not the real-valued roots of polinomial $\det[V(s)\,\mathbf{A}s^2+\mathbf{B}_1(s)\,s+\mathbf{C}_1(s)]$ also lie in the left half plane of s we have recourse to the Nyquist criterion (Mikhailov's criterion, if in the Russian literature).

Prior to proceed with the criterion we will make some rearrangements. First, we introduce the block vector

$$\mathbf{x}=\left|\begin{array}{c}\dot{\mathbf{q}}\\ \hline \mathbf{q}\end{array}\right| \tag{3.13}$$

and the partitioned matrices

$$\mathbf{M}(s)=\left|\begin{array}{c|c}\mathbf{0} & -\mathbf{A}\\ \hline \mathbf{A} & \mathbf{B}(s)\end{array}\right|,\quad \mathbf{N}(s)=\left|\begin{array}{c|c}\mathbf{A} & \mathbf{0}\\ \hline \mathbf{0} & \mathbf{C}(s)\end{array}\right|,$$

and

$$\mathbf{F}(s)=\left|\begin{array}{c}\mathbf{0}\\ \hline \mathbf{G}(s)\end{array}\right|. \tag{3.14}$$

Then the equation (3.8) can be rewritten to give

$$[\mathbf{M}(s)\,s+\mathbf{N}(s)]\,\mathbf{x}(s)=\mathbf{F}(s)\,\mathbf{g}(s). \tag{3.15}$$

According to (3.11)

$$\mathbf{M}(s)=\frac{\mathbf{M}_1(s)}{V(s)},\quad \mathbf{N}(s)=\frac{\mathbf{N}_1(s)}{V(s)},\quad \mathbf{F}(s)=\frac{\mathbf{F}_1(s)}{G_2(s)}, \tag{3.16}$$

where the elements of the matrices $\mathbf{M}_1(s)$, $\mathbf{N}_1(s)$ and $\mathbf{F}_1(s)$ are polynomials in s. With this notation (3.15) reduces to

$$G_2(s)\,[\mathbf{M}_1(s)\,s+\mathbf{N}_1(s)]\,\mathbf{x}(s)=V(s)\,\mathbf{F}_1(s)\,\mathbf{g}(s). \tag{3.17}$$

Location of the roots of polinomial $\det [V(s)\,\mathbf{A}s^2 + \mathbf{B}_1(s)\,s + \mathbf{C}_1(s)]$ is equivalent to the positioning of the roots for

$$\det [\mathbf{M}_1(s)\,s + \mathbf{N}_1(s)] = \tilde{D}(s). \tag{3.18}$$

According to the Nyquist criterion, for the system to be stable the polar plot of (3.18), with zero initial conditions and $s = j\omega$, should vary by $\pi m/2$ as ω varies from 0 to $+\infty$; m is the order of the polynomial (3.18). The same in mathematical form is

$$\Delta \underset{0\leqslant\omega<\infty}{\arg} \tilde{D}(j\omega) = \Delta \underset{0\leqslant\omega<\infty}{\arg} \{\det [j\omega\,\mathbf{M}_1(j\omega) + \mathbf{N}_1(j\omega)]\} = \frac{\pi}{2}\,m. \tag{3.19}$$

The principle of the argument (3.19) can be represented in another form. To this end we designate in (3.15)

$$D(s) = \det [s\mathbf{M}(s) + \mathbf{N}(s)], \tag{3.20}$$

which in view of (3.16) can be written

$$D(s) = \frac{1}{V(s)} \det [s\mathbf{M}_1(s) + \mathbf{N}_1(s)]. \tag{3.21}$$

With the above notation, (3.19) transforms to

$$\Delta \underset{0\leqslant\omega<\infty}{\arg} D(j\omega) = \Delta \underset{0\leqslant\omega<\infty}{\arg} \{\det [j\omega\mathbf{M}_1(j\omega) + \mathbf{N}_1(j\omega)]\}$$

$$-\Delta \underset{0\leqslant\omega<\infty}{\arg} V(j\omega) = \frac{\pi}{2}(m + m_{\mathrm{R}} - m_{\mathrm{L}}), \tag{3.22}$$

where m is the degree of the polynomial $\tilde{D}(s)$; m_{R} and m_{L}' are the number of roots of polynomial $V(s)$ in the right and left half planes, respectively.

From (3.12) we have

$$\det \mathbf{M}(s) = (\det \mathbf{A})^2 \neq 0, \quad \mathbf{M}^{-1}(s) = \left|\begin{array}{c|c} \mathbf{A}^{-1}\mathbf{B}(s)\,\mathbf{A}^{-1} & \mathbf{A}^{-1} \\ \hline -\mathbf{A}^{-1} & \mathbf{0} \end{array}\right|,$$

$$\mathbf{M}^{-1}(s)\,\mathbf{N}(s) = \left|\begin{array}{c|c} \mathbf{A}^{-1}\mathbf{B}(s) & \mathbf{A}^{-1}\mathbf{C}(s) \\ \hline -\mathbf{I} & \mathbf{0} \end{array}\right|, \quad \mathbf{M}^{-1}(s)\,\mathbf{F}(s) = \left|\begin{array}{c} \mathbf{A}^{-1}\mathbf{G}(s) \\ \hline \mathbf{0} \end{array}\right|. \tag{3.23}$$

Invoking the equations of (3.23) we will transform (3.22) considering that det A is a real number, for which

$$\Delta \underset{0\leqslant\omega<\infty}{\arg} \{(\det \mathbf{A})^2\} = 0.$$

Then

$$\underset{0\leqslant\omega<\infty}{\Delta \arg} D(j\omega) = \underset{0\leqslant\omega<\infty}{\Delta \arg}\{(\det \mathbf{A})^2\} + \underset{0\leqslant\omega<\infty}{\Delta \arg}\{\det [j\omega\mathbf{I} + \mathbf{M}^{-1}(j\omega)\,\mathbf{N}(j\omega)]\}$$

$$= \underset{0\leqslant\omega<\infty}{\Delta \arg}\{\det [j\omega\mathbf{I} + \mathbf{M}^{-1}(j\omega)\,\mathbf{N}(j\omega)]\} = \frac{\pi}{2}(m + m_{\mathrm{R}} - m_{\mathrm{L}}). \tag{3.24}$$

Transform the matrix $\mathbf{M}^{-1}(j\omega)\,\mathbf{N}(j\omega)$, at a fixed ω, to a diagonal form

$$\mathbf{L}(j\omega)\,\mathbf{M}^{-1}(j\omega)\,\mathbf{N}(j\omega)\,\mathbf{R}(j\omega) = \Lambda(j\omega), \tag{3.25}$$

where $\mathbf{L}(j\omega)$ and $\mathbf{R}(j\omega)$ are the matrices compiled of the left and right eigenvectors of the matrix $\mathbf{M}^{-1}(j\omega)\,\mathbf{N}(j\omega)$.

The thus obtained diagonal matrix

$$\Lambda(j\omega) = \mathrm{diag}\,[\Lambda_1(j\omega),\ \Lambda_2(j\omega),\ \ldots,\ \Lambda_{2n}(j\omega)]$$

contains in its main diagonal the eigenvalues of the matrix $\mathbf{M}^{-1}(j\omega)\,\mathbf{N}(j\omega)$. Since the coefficients of matrices $\mathbf{A}$, $\mathbf{B}(\omega)$ and $\mathbf{C}(\omega)$ are real numbers, to each element $\Lambda_i(j\omega)$ and vectors $\mathbf{L}_i(j\omega)$ and $\mathbf{R}_i(j\omega)$ we may put in correspondence the complex conjugate elements and vectors. Hence we may write

$$\begin{aligned}
\Lambda_i(j\omega) &= \lambda_{Ri}(\omega) + j\lambda_{Ii}(\omega),\\
\Lambda_{i+1}(j\omega) &= \overline{\Lambda_i(j\omega)} = \lambda_{Ri}(\omega) - j\lambda_{Ii}(\omega),\\
\mathbf{L}_i(j\omega) &= \mathbf{l}_{Ri}(\omega) + j\mathbf{l}_{Ii}(\omega),\\
\mathbf{L}_{i+1}(j\omega) &= \overline{\mathbf{L}_i(j\omega)} = \mathbf{l}_{Ri}(\omega) - j\mathbf{l}_{Ii}(\omega),\\
\mathbf{R}_i(j\omega) &= \mathbf{r}_{Ri}(\omega) + j\mathbf{r}_{Ii}(\omega),\\
\mathbf{R}_{i+1}(j\omega) &= \overline{\mathbf{R}_i(j\omega)} = \mathbf{r}_{Ri}(\omega) - j\mathbf{r}_{Ii}(\omega).
\end{aligned} \tag{3.26}$$

The matrices $\mathbf{R}(j\omega)$ and $\mathbf{L}(j\omega)$ are mutually orthogonal, that is

$$\mathbf{R}(j\omega)\,\mathbf{L}(j\omega) = \mathbf{L}(j\omega)\,\mathbf{R}(j\omega) = \mathbf{I}. \tag{3.27}$$

On account of (3.25), (3.26) and (3.27) the expression (3.24) transforms to

$$\underset{0\leqslant\omega<\infty}{\Delta \arg} D(j\omega) = \underset{0\leqslant\omega<\infty}{\Delta \arg}\{\det [j\omega\mathbf{I} + \Lambda(j\omega)]\} = \underset{0\leqslant\omega<\infty}{\Delta \arg}\Big\{\prod_{i=1}^{2n} [j\omega + \Lambda_i(j\omega)]\Big\}$$

$$= \sum_{i=1}^{2n} \underset{0\leqslant\omega<\infty}{\Delta \arg}\,(j\omega + \Lambda_i(j\omega))$$

$$= \sum_{i=1}^{n} \underset{0\leqslant\omega<\infty}{\Delta \arg}\{(j\omega + \lambda_{Ri}(\omega) + j\lambda_{Ii}(\omega)(j\omega + \lambda_{Ri}(\omega) - j\lambda_{Ii}(\omega))\}.$$

Introducing the notations

$$\lambda_{Ri}^2(\omega) + \lambda_{Ii}^2(\omega) = \Omega_i^2(\omega),$$
$$2\lambda_{Ri}(\omega) = 2\zeta_i(\omega)\,\Omega_i(\omega), \tag{3.28}$$

we finally obtain that for the system to be stable it will suffice that

$$\underset{0\leqslant\omega<\infty}{\Delta \arg} D(j\omega) = \sum_{i=1}^{n} \underset{0\leqslant\omega<\infty}{\Delta \arg} D_i(j\omega) = \frac{\pi}{2}(m + m_R - m_L), \tag{3.29}$$

where

$$D_i(j\omega) = (j\omega)^2 + 2\zeta_i(\omega)\,\Omega_i(\omega)\,(j\omega) + \Omega_i^2(\omega).$$

Obviously,

$$\operatorname{Re} D_i(j\omega) = \Omega_i^2(\omega) - \omega^2$$

and

$$\operatorname{Im} D_i(j\omega) = 2\omega\zeta_i(\omega)\,\Omega_i(\omega). \tag{3.30}$$

The polar plot of $D_i(j\omega)$ is shown in Fig. 3.4. For all the plotted curves we have

$$\underset{0\leqslant\omega<\infty}{\Delta \arg} D_i(j\omega) = \frac{\pi}{2}\cdot 2.$$

The variation of the argument of $D_i(j\omega)$ as ω varies from zero to infinity may be evaluated with reference to the plots of $\Omega_i(\omega)$

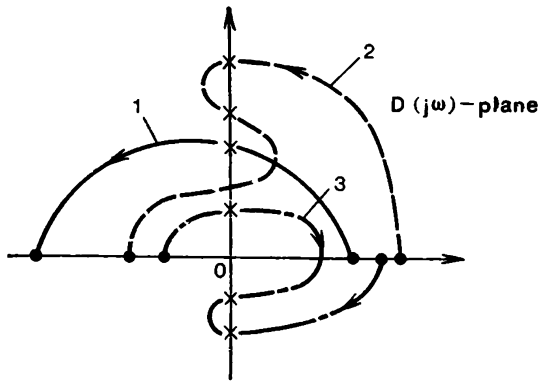


Fig. 3.4 Nyquist loci for function $D_i(j\omega)$

and $\zeta_i(\omega)$ traced as functions of ω. The curves of these functions corresponding to the polar plots of Fig. 3.4 are depicted in Fig. 3.5. Figure 3.6 illustrates the frequencies ω satisfying the equations $\Omega_i(\omega) - \omega = 0$ (denoted by crosses) and $\zeta_i(\omega) = 0$ (heavy points) for the three polar plots shown by curves 1, 2 and 3.

Figures 3.4—3.5 illustrate the procedure which may be employed to evaluate the variation of the argument of $D_i(j\omega)$. First, one should

plot functions $\Omega_i(\omega)$ and $\zeta_i(\omega)$ and mark by crosses the values of ω that satisfy $\Omega_i(\omega) = \omega$ and by points the roots of $\zeta_i(\omega) = 0$ (see, for example, Fig. 3.6). Then two adjacent crosses or points should be deleted on axes *2* and *3* in Fig. 3.6. These values are underlined twice on the figure. The number of segments produced on the positive half axis $0 \leqslant \omega < \infty$ by the retained crosses and points when

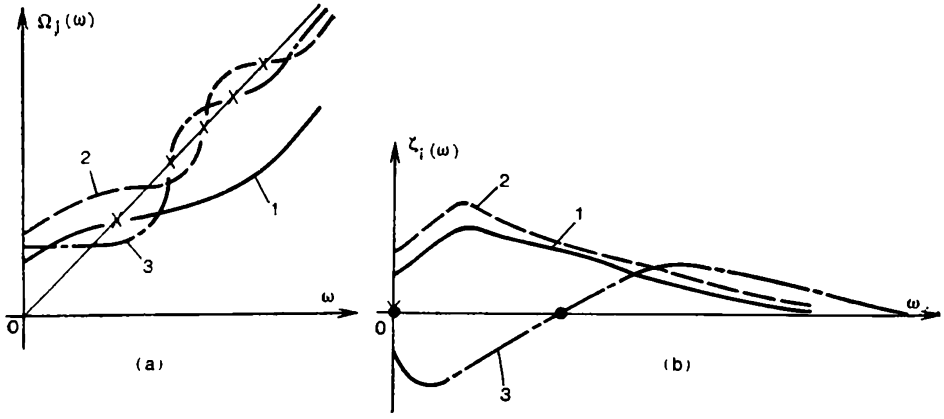


Fig. 3.5 Plots of functions (a) $\Omega_i(\omega)$ and (b) $\zeta_i(\omega)$ for three types of locus

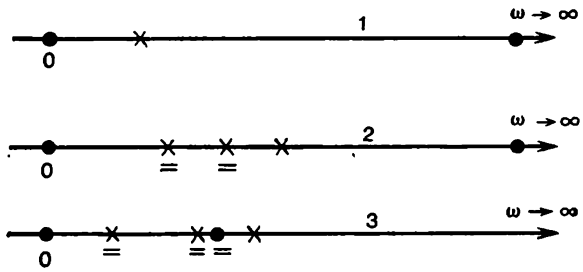


Fig. 3.6 Frequencies satisfying $\Omega_i(\omega) - \omega = 0$ and $\zeta_i(\omega) = 0$ for three loci

multiplied by $\pi/2$ yields, finally, the sought variation of the argument for ω varied from 0 to infinity. Note, that if $\zeta_i(\omega)$ is positive on the first turning of $\Omega_i(\omega) - \omega$ to zero, then the variation of the argument is also positive, otherwise it is negative. The case when $\Omega_i(\omega) - \omega$ and $\zeta_i(\omega)$ turn to zero simultaneously corresponds to a natural mode of the system.

3.3 FREQUENCY RESPONSES OF ROBOT ACTUATOR CONTROL SYSTEM

Consider the frequency responses (FR) of a robot multivariable control system written in terms of the spectral frequency responses. In view of (3.25) and (3.26) the equation (3.15) is

$$[s\mathbf{I} + \mathbf{\Lambda}(s)]\,\mathbf{L}(s)\,\mathbf{x}(s) = \mathbf{L}(s)\,\mathbf{M}^{-1}(s)\,\mathbf{F}(s)\,\mathbf{g}(s) = \mathbf{S}(s)\,\mathbf{g}(s), \quad (3.31)$$

where $\mathbf{S}(s) = \mathbf{L}(s)\,\mathbf{M}^{-1}(s)\,\mathbf{F}(s)$ and $s = j\omega$.
Designate

$$\mathbf{y}(s) = \mathbf{L}(s)\,\mathbf{x}(s), \quad \mathbf{M}^{-1}(s)\,\mathbf{F}(s)\,\mathbf{g}(s) = \mathbf{f}(s), \quad (3.32)$$

then we may write

$$[s\mathbf{I} + \mathbf{\Lambda}(s)]\,\mathbf{y}(s) = \mathbf{L}(s)\,\mathbf{f}(s). \quad (3.33)$$

Since the first equation in (3.32) can be rewritten as

$$\mathbf{x}(s) = \mathbf{R}(s)\,\mathbf{y}(s)$$

we obtain

$$\mathbf{x}(s) = \mathbf{R}(s)\,[s\mathbf{I} + \mathbf{\Lambda}(s)]^{-1}\,\mathbf{L}(s)\,\mathbf{f}(s). \quad (3.34)$$

The matrix $s\mathbf{I} + \mathbf{\Lambda}(s)$ is diagonal, therefore in scalar form the equation (3.34) becomes

$$x_i(s) = \sum_{k=1}^{2n} \sum_{j=1}^{2n} R_{ik}(s) \frac{1}{s+\Lambda_k(s)} L_{kj}(s)\, f_j(s). \quad (3.35)$$

For $s = j\omega$, we can deduce the frequency characteristic corresponding to (3.35). Invoking (3.26) and (3.28) we can obtain the expression of the form

$$\tilde{\Phi}_{ij}(j\omega) = \sum_{k=1}^{2n} \frac{R_{ik}(j\omega)\,L_{kj}(j\omega)}{j\omega + \Lambda_k(j\omega)}$$

$$= \sum_{k=1}^{n} \frac{h_{ijk}(\omega)\,[\tau_{ijk}(\omega) j\omega + 1]}{(j\omega)^2\,T_k^2(\omega) + 2(j\omega)\,T_k(\omega)\,\zeta_k(\omega) + 1}, \quad (3.36)$$

where

$$h_{ijk}(\omega) = [2/\Omega_k^2(\omega)]\,\{[r_{Rik}(\omega)\,l_{Rkj}(\omega) - r_{Iik}(\omega)\,l_{Ikj}(\omega)]\,\lambda_{Rk}(\omega)$$
$$+ [r_{Rik}(\omega)\,l_{Ikj}(\omega) + r_{Iik}(\omega)\,l_{Rkj}(\omega)]\,\lambda_{Ik}(\omega)\}$$
$$\tau_{ijk}(\omega) = \{r_{Rik}(\omega)\,l_{Rkj}(\omega) - r_{Iik}(\omega)\,l_{Ikj}(\omega)\}/$$
$$\{[r_{Rik}(\omega)\,l_{Rkj}(\omega) + r_{Iik}(\omega)\,l_{Ikj}(\omega)]\,\lambda_{Rk}(\omega)$$
$$+ [r_{Rik}(\omega)\,l_{Ikj}(\omega) + r_{Iik}(\omega)\,l_{Rkj}(\omega)]\,\lambda_{Ik}(\omega)\},$$
$$T_k(\omega) = 1/\Omega_k(\omega). \quad (3.37)$$

Denoting

$$U_{ijk}(j\omega) = h_{ijk}(\omega)\,[\tau_{ijk}(\omega)(j\omega)+1], \tag{3.38}$$

we obtain for the frequency characteristic

$$\widetilde{\Phi}_{ij}(\omega) = \sum_{k=1}^{n} \frac{U_{ijk}(j\omega)}{(j\omega)^2 T_k^2(\omega) + 2(j\omega)\zeta_k(\omega) T_k(\omega) + 1}. \tag{3.39}$$

Since

$$\mathbf{M}^{-1}\mathbf{F} = \left[\frac{\mathbf{A}^{-1}\mathbf{G}}{\mathbf{0}}\right] \tag{3.40}$$

all the elements of vector **f** having the subscripts larger than n die out, hence the summation over index j will be from 0 to n and we may write

$$x_i(t) = \sum_{j=1}^{n} \widetilde{\Phi}_{ij}(j\omega) f_j(t), \tag{3.41}$$

where

$f_j(t) = \exp(j\omega t)$.

Introducing the vector

$$\widetilde{\mathbf{U}}_{ik}(j\omega) = [\mathbf{U}_{i1k}(j\omega),\ \mathbf{U}_{i2k}(j\omega),\ \ldots,\ \mathbf{U}_{ink}(j\omega)]. \tag{3.42}$$

we may represent the dynamic diagram of the system in a form displayed on Fig. 3.7, where $(\mathbf{M}^{-1}\mathbf{F})_m$ denotes the mth column of the

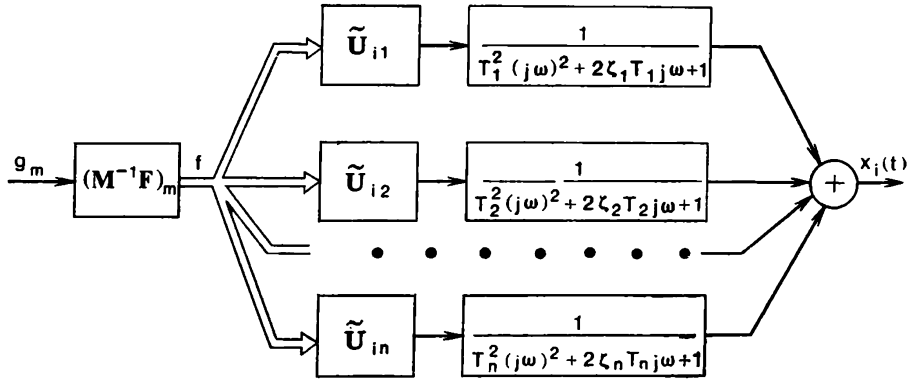


Fig. 3.7 Basic structure of multivariable system with frequency responses

matrix $\mathbf{M}^{-1}\mathbf{F}$ (or $\mathbf{A}^{-1}\mathbf{G}$). This representation lends itself to efficient computer aided synthesis of correcting facilities for the multivariable system.

As has been shown in the course of the above argument, the stability of a multivariable system may well be defined by the spectral frequency responses $\Omega_i(\omega)$ and $\zeta_i(\omega)$, $i = 1, 2, \ldots, n$. The respective synthesis procedures may, thus, be arranged so as to supply the designed system with a desired spectral frequency response. To be sure on this approach, let us ascertain how close the characteristic describes the system performance. Recalling (3.12) to this end, derive the transfer matrix for the close loop system

$$\Phi(s) = [V(s)\,\mathbf{A}s^2 + s\mathbf{B}_1(s) + \mathbf{C}_1(s)]^{-1} V(s) \frac{G_1(s)}{G_2(s)}. \tag{3.43}$$

As will be recalled, an inverse matrix may be approached as $\mathbf{D}^{-1} = \text{adj } \mathbf{D}/\det \mathbf{D}$, where adj $\mathbf{D}$ is the adjugate matrix; hence, we can rewrite (3.43) as

$$\Phi(s) = \frac{V(s)}{\det [V(s)\,\mathbf{A}s^2 + s\mathbf{B}_1(s) + \mathbf{C}_1(s)]} \frac{G_1(s)}{G_2(s)} \mathbf{H}(s), \tag{3.44}$$

where $\mathbf{H}(s)$ is the adjugate matrix of polynomial minors for $V(s)\,\mathbf{A}s^2 + s\mathbf{B}_1(s) + \mathbf{C}_1(s)$.

Labelling the poles of the multivariable system α_i, $i = 1, 2, \ldots, m_1$, (m_1 is the order of the closed loop characteristic polynomial), and the zeros of the system (the roots of the polynomial $V(s)$ of m_2th order) β_j, $j = 1, 2, \ldots, m_2$, reduces (3.44) to

$$\Phi(s) = \frac{K \prod_{j=1}^{m_2} (s - \beta_j)}{\prod_{i=1}^{m_1} (s - \alpha_i)} \frac{G_1(s)}{G_2(s)} \mathbf{H}(s). \tag{3.45}$$

On the other side, as has been shown earlier, (3.44) may be also represented by

$$\begin{aligned} \Phi(s) &= \frac{1}{\det [\mathbf{A}s^2 + s\mathbf{B}(s) + \mathbf{C}(s)]} \frac{G_1(s)}{G_2(s)} \mathbf{H}(s) \\ &= \frac{K_1}{\prod_{i=1}^{n} [s^2 + 2\zeta_i(s)\,\Omega_i(s)\,s + \Omega_i^2(s)]} \frac{G_1(s)}{G_2(s)} \mathbf{H}(s), \end{aligned} \tag{3.46}$$

where K and K_1 stand for some constant coefficients.

Comparing (3.45) and (3.46) indicates that the requirement for the system to possess a desired spectral frequency responses $\Omega_i(\omega)$ and $\zeta_i(\omega)$, $i = 1, 2, \ldots, n$ is equivalent to the desired location of all its poles and zeros. Unfortunately, the spectral frequency responses cannot account for the placement of the multivariable system zeros which are defined by the elements of the polynomial

matrix $\mathbf{H}(s)$, characterizing each of its n^2 channels. This leads to some distortion of the system dynamic properties, specified by the desired characteristics, without exerting any effect on its stability and the pattern of transients (oscillatory or aperiodic).

When the n control systems are treated independently of each other the zeros of $\mathbf{H}(s)$ diagonal elements $h_{ii}(s)$ coincide in part with the poles, simplifying thus the analysis of all the n systems as a whole. In the multivariable system, the roots of the polynomials $h_{ij}(s)$, j, $i = 1, 2, \ldots, n$, receive some increments with respect to their values in an independent n systems analysis. These variations, as is seen from Fig. 3.7, are considered in the multivariable system representation by the elements of matrices $\tilde{\mathbf{U}}_{i,k}$; $i, k = 1, 2, \ldots, n$. These elements bear the information on the cross-coupling in the system and may be employed in attempting to decouple the control systems serving separate joints, when the general objective involves the subsystems being autonomous rather than only a specified performance of the system.

3.4 SYNTHESIS OF ROBOT ACTUATORS WITH SPECTRAL FREQUENCY RESPONSES

Complexity of the designed systems suggests for their synthesis an iterative procedure of several stages with recursive improvement of the mathematical model. As a rule, the synthesis of robot actuator system follows that of noninteracting control systems, which mathematically means zero off-diagonal elements in the matrices $\mathbf{A}$, $\mathbf{B}(s)$, $\mathbf{C}(s)$ and $\mathbf{G}(s)$ of equations (3.15). The design stage is performed by way of synthesis of correcting facilities with well developed methods, say, with the log frequency characteristic approach. This results in the desired spectral frequency responses belonging already to the multivariable system. The characteristics can be readily obtained by representing the frequency responses of separately synthesized systems as

$$\Phi_i(j\omega) = \frac{[Q_i'(j\omega) + K_i'(j\omega)]\, V_{Mi}(j\omega)\, a_{ii}^{-1}}{(j\omega)^2 + 2\zeta_i(\omega)\,\Omega_i(\omega)\, j\omega + \Omega_i^2(\omega)} \tag{3.47}$$

and solving them for $\Omega_i(\omega)$ and $\zeta_i(\omega)$, $i = 1, 2, \ldots, n$. In (3.47), $Q_i'(j\omega)$ and $K_i'(j\omega)$ are the frequency responses of correction performed sequentially and with respect to the control, respectively; a_{ii} is the (i,i)th element of the matrix $\mathbf{A}$.

It should be emphasized that the desired spectral frequency responses are developed so as to meet the requirement to autonomy, i.e. the absence of any cross-coupling between the systems. Then, the performance of the designed systems is analyzed under full consideration of cross couplings, which mathematically implies that the off-

diagonal elements of the matrices **A**, **B** (s), **C** (s) and **G** (s) are no longer zeros.

If the number of the cross couplings is large, it may bring about an unsatisfactory performance and even instability of multivariable system. Hence the second step in the synthesis—to provide the desired performance of the actuator system by choosing suitable correction matrices **Q** (s), **K** (s), and **Z** (s). In the following we will have a closer insight into the problem.

To assure the required dynamic precision for the actuator system, the correction matrices $\mathbf{Q}(s) = k_0\mathbf{Q}_0(s)$ and $\mathbf{K}(s) = \mathbf{K}_\beta s$ will be chosen on the condition that the matrices of frequency characteristics of the open-loop multivariable system and of desired frequency responses for separate control systems should be equal in low frequency band, that is

$$\mathbf{W}(j\omega)|_{\omega\to 0} = \mathbf{W}_{\text{desired}}(j\omega)|_{\omega\to 0}. \tag{3.48}$$

Let us express the matrix $\mathbf{W}(j\omega)|_{\omega\to 0}$ in (3.48) through the matrices of error coefficients $\mathbf{C}_g$ and $\dot{\mathbf{C}}_g$. We will try to achieve this by setting up the matrix of close-loop actuator system in error

$$\mathbf{\Phi}_\varepsilon(s) = [\mathbf{I} + \mathbf{W}(s)]^{-1} \approx \mathbf{C}_g + \dot{\mathbf{C}}_g s. \tag{3.49}$$

Since at low frequencies the following inequality is valid

$$\mathbf{W}_{ij}(j\omega)|_{\omega\to 0} \gg 1, \quad i, j = 1, 2, \ldots, n,$$

from (3.49) we obtain

$$\mathbf{W}^{-1}(j\omega)|_{\omega\to 0} \approx \mathbf{C}_g + \dot{\mathbf{C}}_g s. \tag{3.50}$$

The matrices of coefficients, $\mathbf{C}_g$ and $\dot{\mathbf{C}}_g$, in (3.50) are expressed via the coefficients of matrices **B** and **C** and can be derived by

$$\mathbf{C}_g = \mathbf{\Phi}_\varepsilon(s)|_{s=0}, \quad \dot{\mathbf{C}}_g = \frac{d\mathbf{\Phi}_\varepsilon(s)}{ds}\Big|_{s=0}. \tag{3.51}$$

By substituting into (3.51) the expression for

$$\mathbf{\Phi}_\varepsilon(s) = [\mathbf{A}s^2 + \mathbf{B}(s)s + \mathbf{C}(s)]^{-1}[\mathbf{A}s^2 + \mathbf{B}_F(s)s + \mathbf{C}_F(s)], \tag{3.52}$$

where $\mathbf{B}_F(s)s + \mathbf{C}_F(s) = \mathbf{W}_F(s)$, we arrive at the sought equations

$$\mathbf{C}_g = \mathbf{C}_F^{-1}(0)\,\mathbf{C}_F(0),$$
$$\dot{\mathbf{C}}_g = \mathbf{C}^{-2}(0)[\mathbf{C}(0)\mathbf{B}_F(0) - \mathbf{B}(0)\mathbf{C}_F(0)]. \tag{3.53}$$

Denoting the diagonal matrices for n individually synthesized control systems by $\mathbf{C}'_g$, $\dot{\mathbf{C}}'_g$, $\mathbf{Q}'(s) = k'_0\mathbf{Q}'_0(s)$ and $K'(s) = K'_\beta s$, and substituting the expressions (3.50) into (3.48) and taking account of the matrices $\mathbf{C}'_g$ and $\dot{\mathbf{C}}'_g$ only, we get

$$\mathbf{C}_g^{-1}[\mathbf{K}_0\mathbf{Q}_0(s) + \mathbf{K}_\beta s]|_{s=0} = (\mathbf{C}'_g)^{-1}[\mathbf{K}'_0\mathbf{Q}'_0(s) + \mathbf{K}'_\beta s]|_{s=0}.$$

Letting $\mathbf{Q}_0(0) = \mathbf{Q}_0'(0) = \mathbf{I}$ gives for the above

$$\mathbf{K}_0 = \mathbf{C}_g [\mathbf{C}_g']^{-1} \mathbf{K}_0' \tag{3.54}$$

and

$$\mathbf{K}_\beta = \mathbf{C}_g [\mathbf{C}_g']^{-1} \mathbf{K}_\beta'. \tag{3.55}$$

In a similar manner we can find the expressions for $\mathbf{K}_0$ and $\mathbf{K}_\beta$ taking into consideration matrix $\mathbf{C}_g$. Note that $\mathbf{Q}_0(s)$ may here be put equal to $\mathbf{Q}_0'(s)$ derived in the synthesis of n separate systems, that is,

$$\mathbf{Q}_0(s) = \mathbf{Q}_0'(s). \tag{3.56}$$

Choosing the matrices of coefficients by (3.54) and (3.55) boosts the low-frequency response of the multivariable system with the intention to compensate for the errors due to cross-couplings in low frequency range.

Let us elucidate now the manner in which the specified dynamic properties of the actuator system can be obtained. Since one of the requirements consists in driving the transients to specified patterns decided mainly by the properties of the system near the cutoff frequency, the fitness of the spectral frequency responses with the desired ones may be aimed at in this narrow range. In the synthesis, the model of the system in the form of (3.8) is evaluated for every ω of the range. The fitting of spectral frequency responses derived for each ω may be thought of as synthesis of purely mechanical oscillatory system yielding the required values of both matrices $\mathbf{B}(\omega)$ and $\mathbf{C}(\omega)$, that is the equivalent viscosities and flexibilities in the form of $\Omega_i(\omega)$ and $\zeta_i(\omega)$, $i = 1, 2, \ldots, n$.

Coming over from one frequency to another results in the frequency functions for the elements $b_{ij} = b_{ij}^*(\omega)$ and $c_{ij} = c_{ij}^*(\omega)$, $i, j = 1, 2, \ldots, n$, that is in the desired frequency responses of the elements of matrices $\mathbf{B}(s) = \mathbf{B}^*(s)$ and $\mathbf{C}(s) = \mathbf{C}^*(s)$. Being implemented, these matrices give rise to the required spectral frequency responses. Recalling the dependences (3.3), (3.5) and (3.6) for matrices $\mathbf{B}(s)$, $\mathbf{C}(s)$ and $\mathbf{W}_\Sigma(s)$, the sought matrix $\mathbf{Z}(s)$ becomes

$$\mathbf{Z}(s) = \mathbf{W}_M^{-1}(s) [\mathbf{W}_\Sigma^*(s) - \mathbf{W}_F(s) - \mathbf{W}_M(s) \mathbf{Q}(s)], \tag{3.57}$$

where $\mathbf{W}_\Sigma^*(s)$ is found from (3.5) as

$$\mathbf{W}_\Sigma^*(s) = \mathbf{W}_1^*(s)\, s + \mathbf{W}_2^*(s) = [\mathbf{B}^*(s) - \mathbf{b}]\, s + [\mathbf{C}^*(s) - \mathbf{c}]. \tag{3.58}$$

Consider the evaluation of matrices $\mathbf{B}^*(s)$ and $\mathbf{C}^*(s)$ at first for the case when the correction cross-couplings are absent and matrix $\mathbf{Z}(s)$ is diagonal.

Let, as before, $\Omega_i(\omega)$ and $\zeta_i(\omega)$, $i = 1, 2, \ldots, n$, be the desired spectral frequency response coming as the solutions of the n equations

$$a_{ii}\lambda^2 + b_{ii}(\omega)\lambda + c_{ii}(\omega) = 0, \quad i = 1, 2, \ldots, n, \tag{3.59}$$

and $\widetilde{\Omega}_i(\omega)$ and $\widetilde{\zeta}_i(\omega)$, $i = 1, 2, \ldots, n$, be the spectral frequency responses of the multivariable actuator system obtained as the solutions to

$$\det[\mathbf{A}\widetilde{\lambda}^2 + \mathbf{B}(\omega)\widetilde{\lambda} + \mathbf{C}(\omega)] = 0. \tag{3.60}$$

Then, assuming $\widetilde{\lambda}_i = \lambda_i$, or $\widetilde{\Omega}_i = \Omega_i$, and $\widetilde{\zeta}_i = \zeta_i$, $i = 1, 2, \ldots, n$, we get, equating the respective equations

$$\det(\underline{\mathbf{A}}\lambda^2 + \underline{\mathbf{B}}^*\lambda + \underline{\mathbf{C}}^*) = \prod_{i=1}^{n}(\underline{a}_{ii}\lambda^2 + \underline{b}_{ii}\lambda + \underline{c}_{ii}) = 0, \tag{3.61}$$

where the known quantities are marked with a bar below, and the diagonal matrices $\mathbf{B}^*(s)$ and $\mathbf{C}^*(s)$ are to be found. The equation (3.61) expands out into the set of $2n$ nonlinear algebraic equations for the coefficients of equal powers of λ. Solving these equations by Newton's method yields the values of $\mathbf{B}^*(\omega)$ and $\mathbf{C}^*(\omega)$ and, hence, through (3.57) the matrix of parallel correction $\mathbf{Z}(s)$.

It should be noted, that the spectral frequency responses of independent control systems will be now defined as the solutions to equations of a form

$$a_{ii}^*\lambda^2 + b_{ii}^*\lambda + c_{ii}^* = 0 \tag{3.62}$$

and may markedly differ from $\Omega_i(\omega)$ and $\zeta_i(\omega)$, $i = 1, 2, \ldots, n$.

Obviously, the synthesis procedure should, in general, reconcile the following two contradicting objectives: the synthesis of n independent control systems which may lead to unsatisfactory dynamics for the entire system of actuators, on the one hand, and striving for the required dynamic properties for the multivariable system which may lead even to the loss of stability by some inner loops of the system, on the other. Striving towards this goal we arrive at a problem of optimal design.

Let $\mathbf{x}(s) = [\mathbf{b}^*(s) \mid \mathbf{c}^*(s)]^T$ be the sought $2n$-dimensional vector composed of the diagonal elements of matrices $\mathbf{B}^*(s)$ and $\mathbf{C}^*(s)$, and $\Omega_i(\omega)$ and $\zeta_i(\omega)$, $i = 1, 2, \ldots, n$, be the desired spectral frequency responses and $\widetilde{\Omega}_i(\omega)$ and $\widetilde{\zeta}_i(\omega)$ their counterparts obtained for the multivariable system from (3.60). Let also $\Omega_i^{**}(\omega)$ and $\zeta_i^{**}(\omega)$, $i = 1, 2, \ldots, n$, be the spectral responses of independent control systems and $\Delta\Omega_i^{**}(\omega)$ and $\Delta\zeta_i^{**}(\omega)$ be their tolerable va-

riations engineered to improve the behavior of the entire actuator system and such that to keep the system within the stability limits. Then the design objective may be stated as follows. To find the value of the vector $\mathbf{x} = \mathbf{x}^*$ which minimizes, for every ω, the nonlinear function

$$y(\mathbf{x}) = \sum_{i=1}^{n} \{[\tilde{\Omega}_i(\omega) - \Omega_i(\omega)]^2 + [\tilde{\zeta}_i(\omega) - \zeta_i(\omega)]^2\} \tag{3.63}$$

subject to the given constraints

$$|\Omega_i^* - \Omega_i^{**}| \leqslant \Delta\Omega_i^{**}, \quad |\zeta_i^* - \zeta_i^{**}| \leqslant \Delta\zeta_i^{**}, \tag{3.64}$$

$i = 1, 2, \ldots, n.$

In these constraints, $\Omega_i^*(\omega)$ and $\zeta_i^*(\omega)$ stand for the spectral frequency responses of individual control systems.

With this formulation the problem of nonlinear programming turns into a problem of conditional optimization which can be solved on a computer by a variety of numerical computation techniques.

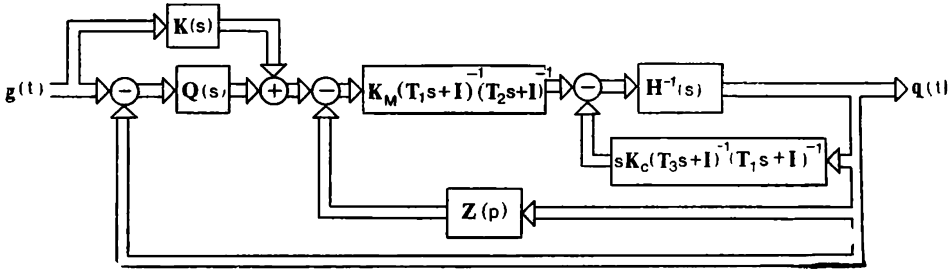


Fig. 3.8 Block diagram of the drive control system from the example in the text

The suggested linear programming approach to the synthesis of complex robot actuator systems brings about the correction laws even for the matrix $\mathbf{Z}(s)$ containing off-diagonal elements, i.e. involving the correcting cross-coupling. The dimension of the vector $\mathbf{x}$ for this case respectively increases to include all the nonzero elements of the matrices $\mathbf{B}(s)$ and $\mathbf{C}(s)$. Prior to run the problem on a computer the designer may state those cross-couplings which he believes are technically viable, by pointing their inputs and outputs on the block diagram. With this orientation the process of computer synthesis of the correction laws may be arranged along the lines of gradual complication if the set of correcting signals got from the previous run cannot yield any satisfactory realization to correcting facilities. To have a closer insight into how the suggested procedure works consider an example of such a correction law synthesis.

Illustrative example. Consider a block diagram of the robot control system (Fig. 3.8) whose kinematic diagram is presented in

Fig. 3.9. The displayed robot has three degrees of freedom ($n = 3$). $\mathbf{T}_1$, $\mathbf{T}_2$, $\mathbf{T}_3$, $\mathbf{K}_M$, and $\mathbf{K}_F$ stand for the diagonal matrices characterizing the actuator section of the system without any correction. Matrix

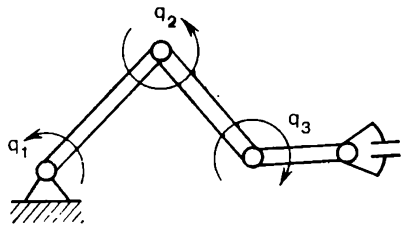


Fig. 3.9 Kinematic diagram of robot actuator system

A, having its off-diagonal elements, represents the inertial properties of the actuator, matrices **Q** (s), **K** (s) and **Z** (s) denote the correcting law to be found, and **I** is the identity matrix. The parameters describing the nonvariable sections of the systems are arrayed below.

Connectivity	T_1	T_2	T_3	K_M	K_F
q_1	0.003	0.02	0.00086	1.0	72
q_2	0.003	0.0167	0.00092	1.0	25.6
q_3	0.003	0.027	0.00094	1.0	14.5

With the matrices **HA**, **b**, and **c** specified as follows

$$\mathbf{A} = \begin{vmatrix} 1.8 & 0.95 & 0.24 \\ 0.95 & 0.53 & 0.14 \\ 0.24 & 0.14 & 0.042 \end{vmatrix}, \quad \mathbf{c} = \begin{vmatrix} -17 & -8.6 & -2.1 \\ -8.6 & -8.6 & -2.1 \\ -2.1 & -2.1 & -2.1 \end{vmatrix},$$

$$\mathbf{b} = 0, \quad \mathbf{H} = \mathbf{A}s^2 + \mathbf{b}s + \mathbf{c}$$

the synthesis of n individual control systems, oriented on their precision, yielded the following correction matrices

$$\mathbf{K}'(s) = 0, \quad \mathbf{Z}'(s) = 0, \quad \mathbf{Q}'(s) = \text{diag } [1480, 520, 290].$$

The spectral frequency responses for both the corrected independent systems (matrix **A** diagonal) and multivariable control system are shown in Fig. 3.10. It is seen that though the system retained its stability the dynamic behavior is substantially distorted by cross-coupling—the system became more oscillatory, which is decided by the lowest pair of curves $\Omega_i(\omega)$ and $\zeta_i(\omega)$, the response became slower, this is decided by the characteristics positioned as the highest on the plot. The pattern in the whole indicates the need to additional correction to compensate for the generated distortions.

The respective synthesis of multivariable control system yields the matrix $\mathbf{Z}(s)$ in the form

$$\mathbf{Z}(s) = \text{diag}\,[-0.8\,(1 + 0.12s),\ -0.9\,(1 + 0.04s),\ 16\,(1 + 0.1s)].$$

The matrices $\mathbf{Q}(s) = \mathbf{Q}'(s)$ and $\mathbf{K}(s) = \mathbf{K}'(s)$ were obtained earlier. The synthesized correction laws are realized on microprocessors and mini-computers making up the robot control system.

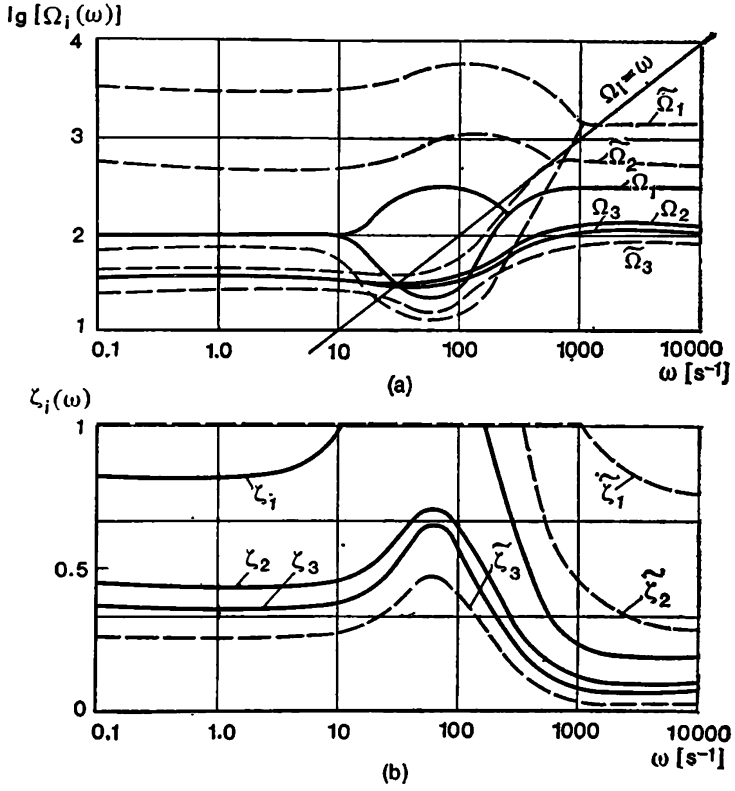


Fig. 3.10 Frequency responses of the robot control system for (a) $\Omega_i(\omega)$, (b) $\zeta_i(\omega)$

List of Symbols

$D = d/dt$	differential operator
s	Laplace variable
ω	angular frequency
$\mathbf{q}(t)$	vector of the generalized coordinates
$\mathbf{g}(t)$	vector of controls
$\Omega_i(\omega)$ and $\zeta(\omega_i)$	spectral frequency responses

4 Path Synthesis for Robot Linkage in Crowded Workspace

V. L. Generozov

4.1 GENERAL

This chapter is devoted to the computer aided path synthesis of the robot arm linkage in an environment with obstacles. The considered approach is based on a linearized kinematic model of the linkage and uses small increments of the generalized coordinates which are computed sequentially by solving a series of linear programming (LP) problems. The constraints to the LP problems are set up so that the linkage could avoid the obstacles; they are formulated with the use of parametric linear programming, as are done Chebyshev's distances from the points on manipulator links to the obstacles of the convex polyhedron type, and the subgradients of these distances. The devised algorithm is demonstrated on a few numerically modeled examples.

The suggested method is sensible for the manipulators offering positive manoeuvrability*. It can be implemented to the path synthesis and motion guidance of supervisory-controlled and artificial-intelligence robotic manipulators. For industrial (computer augmented) robots, the method can supply the computer designed programmed displacements to be fulfilled in an environment heavily crowded with obstacles.

4.2 MODELING ARM LINKAGE AND ENVIRONMENTAL WORKSPACE

We assume a geometric model of the manipulator to consist of a spatial NL-bar open polygone linkage with broken-line links L_l in the chain of $l = 1, 2, \ldots, NL$ links, which is equivalent to a representation by a finite system of cylinders.

We shall think of the workspace environment as consisting of the objects composed of convex polyhedrons. This representation of the environment is fairly general since most part of real bodies can be approximated by the objects of this class.

---

* That is, having redundant degrees of freedom $M > n$, with the manoeuvrability $m = M - n$ where n is the dimension of workspace.—*Translator's note.*

As the *configuration of the robot linkage* $\mathbf{q} = [q_1, q_2, \ldots, q_i, \ldots, q_N]^T$ we shall refer to a vector of the values of the generalized coordinates q_i of robot's joints, $i = \overline{1, N}$; N is the number of joints; and T indicates transposition.

A configuration of the linkage is deemed *admissible* if:

(i) the values of the generalized coordinates satisfy the constraints placed by the construction of the linkage, namely,

$$q_{i,\min} \leqslant q_i \leqslant q_{i,\max}, \quad i = 1, 2, \ldots, N;$$

(ii) the links of the chain have no intersections with the obstacles, or, more precise, are distanced to them not closer than a certain tolerable value (the notion of distance to be defined later).

The *state* (location and orientation) *of the hand* is defined by the matrix $\mathbf{T}^N = [\mathbf{l}^N | \mathbf{p}^N]$ of size 3×4 partitioned into the column-vector $\mathbf{p}^N$ and the 3×3 orthogonal submatrix $\mathbf{l}^N$ which respectively define the origin and orientation of the hand-tied reference frame with respect to the inertial coordinate system.

Let the initial configuration of the linkage $\mathbf{q}_s$ and the terminal state of the hand $\mathbf{q}_f$ be given. It is required to determine the sequence of admissible configurations for the linkage $\mathbf{q}_s = \mathbf{q}[1] \rightarrow \ldots \rightarrow \mathbf{q}[2] \rightarrow \ldots \rightarrow \mathbf{q}[k] \rightarrow \ldots \rightarrow \mathbf{q}[M] = \mathbf{q}_f$ (M is the number of nodal points on the path) quiding the hand from its initial state $\mathbf{T}_s^N$, defined by the configuration of the chain $\mathbf{q}_s$, to the terminal state $\mathbf{T}_f^N$. The hand should be moved through the workspace along a path which is close to the given linear segment $\mathbf{p}_f^N - \mathbf{p}_s^N$ (the vectors $\mathbf{p}_s^N$ and $\mathbf{p}_f^N$ define respectively the initial and terminal state of the hand). The condition for the links L_l, $l = \overline{1, NL}$, to avoid obstacle C may be stated by

$$d(L_l, C) \geqslant r_l, \quad l = 1, 2, \ldots, NL, \tag{4.1}$$

where

$$d(L_l, C) = \min_{\mathbf{x} \subset L_l} d(\mathbf{x}, C) \tag{4.2}$$

is the distance from the link L_l to the closed convex set C, and r_l is a given value of minimally tolerable distance to the obstacle for the link L_l; $d(\mathbf{x}, C)$ implies

$$d(\mathbf{x}, C) = \min_{\mathbf{y} \subset C} \| \mathbf{x} - \mathbf{y} \| \tag{4.3}$$

that is the distance from point $\mathbf{x}$ to C specified as the norm $\| \ \|$ in space $\mathcal{R}^n$ of dimension n. In particular the expressions

$$\| \mathbf{x} \|_\infty = \max_{1 \leqslant i \leqslant n} |x_i| \quad \text{and} \quad \| \mathbf{x} \|_2 = \left(\sum_{i=1}^{n} x_i^2 \right)^{1/2}$$

generate, respectively, the Chebyshev and Euclidean norms for vector $\mathbf{x}$ related by the inequality (the equivalence of norm in $\mathcal{R}^n$)

$$\| \mathbf{x} \|_\infty \leqslant \| \mathbf{x} \|_2 \leqslant \sqrt{n} \| \mathbf{x} \|_\infty. \tag{4.4}$$

Various approaches to the problem of path synthesis, to guide the manipulator linkage away from the obstacles, based on a linearized kinematic model of the linkage are covered, for example, in [7, 20, 26].

Reference [7], in particular, suggests an algorithm of path synthesis for the planar linkage in an environment consisting of convex polygons; the algorithm builds up with LP technique, reasonably accounting for the planarity of the problem.

The path synthesis approach considered in this chapter is a further extension of the discrete algorithm discussed in [3, 43] in which linkage displacement is constructed successively by the use of LP. Main emphasis in the discussion will be placed on how to form the constraints due to the obstacles and to model the algorithm of the synthesis subject to these constraints.

To formulate the constraints, caused by the obstacles, in the form linear in the increments of the generalized coordinates, we assume that at the kth step of the synthesis we get for the link L_l

$$\mathrm{d}^k (L_l, C) = \mathrm{d} (\mathbf{x}_l [k], C).$$

Here $\mathbf{x}_l [k]$ is the extremum point on the link L_l, i.e., the point for which the distance to the closed convex set C is minimal of all the points in L_l.

As will be recalled (see, for example, [45]), the function of distance $\mathrm{d} (\mathbf{x}, C)$ from a point $\mathbf{x}$ to the convex set C in $\mathcal{R}^n$ is a convex function of $\mathbf{x}$ and, hence, satisfies the so-called subgradient inequality

$$\mathrm{d} (\mathbf{z}, C) \geqslant \mathrm{d} (\mathbf{x}, C) + \langle \mathbf{x}^*, \mathbf{z} - \mathbf{x} \rangle, \text{ for all } \mathbf{z} \subset \mathcal{R}^n, \tag{4.5}$$

where

$$\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{n} x_i y_i.$$

The vector $\mathbf{x}^* \subset \mathcal{R}^n$ is called the subgradient or supporting vector of the convex function $\mathrm{d} (\mathbf{x}, C)$ at point $\mathbf{x}$. We wish to recall that the notion of subgradient is a natural generalization of that of the gradient for nonsmooth convex functions. The set of all subgradients of a convex function $f (\mathbf{x})$ at $\mathbf{x}$ is referred to as the subdifferential of $f (\mathbf{x})$ at $\mathbf{x}$ and is denoted $\partial f (\mathbf{x})$ [45]. If $f (\mathbf{x})$ has its derivatives at $\mathbf{x}$, then $\partial f (\mathbf{x})$ consists of the sole vector—its own gradient $f' (\mathbf{x})$.

With the above nomenclature, if $\Delta \mathbf{x}_l [k]$ is the increment vector of coordinates for the point $\mathbf{x}_l [k]$ then by (4.5) we have

$$\mathrm{d} (\mathbf{x}_l [k] + \Delta \mathbf{x}_l [k], C) \geqslant \mathrm{d} (\mathbf{x}_l [k], C) + \langle \nabla \mathrm{d}_l^k, \Delta \mathbf{x}_l [k] \rangle, \tag{4.6}$$

where $\nabla \mathrm{d}_l^k$ is the subgradient of the function $\mathrm{d} (\mathbf{x}, C)$ at $\mathbf{x}_l [k]$.

Require next that the right-hand side of (4.6) satisfy the inequality

$$\mathrm{d}\,(\mathbf{x}_l\,[k],\ C) + \langle \nabla \mathrm{d}_l^k,\ \Delta\mathbf{x}_l\,[k]\rangle \geqslant r_l. \tag{4.7}$$

From (4.6) and (4.7) it follows then

$$\mathrm{d}\,(\mathbf{x}_l\,[k] + \Delta\mathbf{x}_l\,[k],\ C) \geqslant r_l. \tag{4.8}$$

Or in words: from the conditions (4.7) on the increments of coordinates $\Delta\mathbf{x}_l\,[k]$ at the extremal points it follows that the new positions (at step $k+1$) of the former extremal points will satisfy (4.8). Note that for a separate link (linear bar) the condition (4.8) does not necessarily imply $\mathrm{d}_l^{k+1}\,(L_l,\ C) \geqslant r_l$. Particularly, for planar motion, consider a bar revolving about an axis passing through the extremal point $\mathbf{x}_l\,[k]$. This revolution may decrease the distance between L_l and C in the sense of (4.2) in spite of $\Delta\mathbf{x}_l\,[k] = 0$ in this case.

Still, as the simulation of the algorithm has shown, formulating the constraints by the system of inequalities (4.7), corresponding to the chain-connected links L_l, $l = \overline{1, NL}$, yields satisfactory results in the sense of preserving the conditions (4.1).

Substituting the expressions for small increments

$$\Delta\mathbf{x}_l\,[k] = \mathbf{J}_l\,[k]\ \Delta\mathbf{q}\,[k]$$

into the inequalities (4.7) we write them in a form linear in the vector of increments of the generalized coordinates $\Delta\mathbf{q}\,[k]$, viz.,

$$-\langle \nabla \mathrm{d}_l^k,\ \mathbf{J}_l\,[k]\ \Delta\mathbf{q}\,[k]\rangle \leqslant \mathrm{d}\,(\mathbf{x}_l\,[k],\ C) - r_l,\ l = \overline{1, NL}. \tag{4.9}$$

Here, $\mathbf{J}_l\,[k]$ is the Jacobian matrix of the mapping $\mathbf{x}_l\,[k] = \mathbf{F}\,(\mathbf{q})$, otherwise labelled the sensitivity matrix at point $\mathbf{x}_l\,[k]$.

In some particular cases, the constraints due to obstacles can be set up in a more simple form. So, if it is required that a certain point of the linkage, say one of its joints, (vertex of the open polygon) be kept away from the table plane, that is the horizontal plane $0X_1X_2$, by a distance not smaller than a given value r, then the respective constraint may be written as

$$-\Delta x_3\,[k] \leqslant x_3[k] - r, \tag{4.10}$$

where

$$\Delta x_3\,[k] = \sum_{j=1}^{N} J_{3j}\,[k]\ \Delta q_j\,[k]$$

is the increment of the coordinate x_3 for the given point, and $\{J_{3j}\}$ $j = \overline{1, NL}$, are the elements in the third row of the sensitivity matrix for this point.

To generate the constraints (4.9) at each step of the synthesis we have to compute for each link L_l its distance $\mathrm{d}\,(L_l,\ C)$ to the obstacle C, the coordinates of the extremal point $\mathbf{x}_l\,[k]$, the sensitivity

matrix for this point $\mathbf{J}_l[k]$ and the subgradient (or gradient) ∇d_l^k of the distance d ($\mathbf{x}$, C) at the link point $\mathbf{x}_l[k]$. This geometric problem of computing distances and the related variables may be run easier or become more complicated depending on the manner in which the distance is specified and the class of objects to which the obstacles belong; in the general case of arbitrary convex objects we arrive at a problem of convex programming.

Consider the evaluation of the distance, defined in the sense of Chebyshev's norm in finite dimensional space, for a model of manipulator linkage in a form of a spatial open polygon with vertices $\mathbf{v}^1, \mathbf{v}^2, \ldots, \mathbf{v}^{NL+1}$ and the obstacles represented by convex polyhedrons specified by the matrices of their faces. This statement of the problem enables a simple and efficient algorithm to be built for it with the use of finite computational methods of LP, parametric LP in particular. In view of the norm equivalence relationship (4.4) any moving away from an obstacle in the sense of the Chebyshev norm, $\| \ \|_\infty$, implies the respective distancing in the sense of the Euclidean norm. The distance function so designed is, as a rule, nondifferentiable at the extremal points. The algorithm discussed below determines the distances and the subdifferentials of distances at the extremal points, which are convex polyhedral sets.

4.3 EVALUATING DISTANCES AND THEIR SUBDIFFERENTIALS

First of all we reduce the problem to a succession of linear programs. On these lines, we will specify each convex polyhedron C by the $m \times n_1$ matrix $\mathbf{F} = [\mathbf{H} \vdots \mathbf{D}]$ describing its faces, in which m rows correspond to the number of faces, and $n_1 = n + 1$ for n-dimensional workspace; $\mathbf{H} = \{h_{ij}\}$ is the $m \quad n$ matrix whose rows are the coordinates of unit outer normals to the faces; and $\mathbf{D} = [d_1, d_2, \ldots, d_m]^T$ is the vector of Euclidean distances of the faces from the origin.

With the above notation, the points of a convex polyhedron represent the set of solutions for the system of linear inequalities

$$\mathbf{Hy} \leqslant \mathbf{D}. \tag{4.11}$$

The problem of finding the distance from a point x to the convex polyhedron C in the sense of the Chebyshev norm, that is, Chebyshev approximating x by the solutions of the inequalities (4.11), may be stated as:

find

$$\min_{y} \max_{1 \leqslant i \leqslant n} |x_i - y_i|$$

subject to (4.11)

can be reduced to the equivalent LP problem [15].

Representing the points on the link L_l parametrically by $\mathbf{x} = \mathbf{v}^l + t\mathbf{w}^l$,

where $\mathbf{w}^l = \mathbf{v}^{l+1} - \mathbf{v}^l$ and $0 \leqslant t \leqslant 1$, we arrive at the following form, representing the distance in terms of (4.2),

$$d(L_l, C) = \min_{0 \leqslant t \leqslant 1} \| \mathbf{v}^l + t\mathbf{w}^l - \mathbf{y} \|_\infty, \quad \mathbf{Hy} \leqslant \mathbf{D}. \tag{4.12}$$

For the model of the linkage having NL links L_l, each of which is given by its outermost points $\mathbf{v}^l$ and $\mathbf{v}^{l+1}$, we are to solve NL problems of the type (4.12).

Denoting the parameter t by x_{n+1} and introducing the additional variable x_{n+2}, which represents the value of the objective function, reduces the problem (4.12) to the following linear program in $n + 2$ variables and $m + 1 + 2n$ constraints:

minimize

x_{n+2}

subject to

$$\begin{gathered}
\sum_{i=1}^{n} h_{ji} x_i \leqslant d_j, \quad j = \overline{1, m}, \quad x_{n+1} \leqslant 1, \\
x_i - w_i^l x_{n+1} - x_{n+2} \leqslant v_i^l, \quad i = \overline{1, n}, \\
-x_i + w_i^l x_{n+1} - x_{n+2} \leqslant -v_i^l, \quad i = \overline{1, n}.
\end{gathered} \tag{4.13}$$

Without any loss of generality we may assume $\mathbf{x}_i \geqslant 0$ for all $i = \overline{1, n+2}$. It will be well advised to solve the problem (4.13) by the dual simplex method since the initial basic solution is dual-feasible basic solution.

The problem (4.12) can be approached in another way. Firstly, one should evaluate a law governing the variation of the distance from $\mathbf{x}$ on L_l to C as a function of the parameter t. The result in this case will be a parametric linear program with a parameter dependent vector of constraints [5, 15]

$$\begin{gathered}
z^l = x_{n+1}, \quad z^l \to \min, \\
\sum_{i=1}^{n} h_{ji} x_i \leqslant d_j, \quad j = \overline{1, m}, \\
x_i - x_{n+1} \leqslant v_i^l + t w_i^l, \quad i = \overline{1, n}, \\
-x_i - x_{n+1} \leqslant -v_i^l - t w_i^l, \quad i = \overline{1, n}, \\
\mathbf{x} \geqslant 0, \quad 0 \leqslant t \leqslant 1, \quad \mathbf{x} = [x_1, \ldots, x_{n+1}]^T.
\end{gathered} \tag{4.14}$$

A solution of the problem (4.12) will be then in the form of $\min_{0 \leqslant t \leqslant 1} z^l(t)$ which can be readily found keeping in sight that $z^l(t)$ is a convex piecewise-linear function of the parameter t. Hence,

for a chain of NL links one obtains NL parametric LP problems (4.14) consecutively connected in that the final solution, at $t = 1$, of the problem for the link L_l will be the initial one (at $t = 0$), for the link L_{l+1}, $l = \overline{1,NL-1}$. The advantage of this parametric LP approach is in that the total number of iterations reduces because the information on the L_l link solution enters the solution for the link L_{l+1}. Besides, the generated set of parametric functions $z^1(t)$, $z^2(t)$, ., $z^{NL}(t)$ describes how the distance toward C changes for all points along the linkage. Moreover, parametric programming elucidates certain features of the problem under consideration, for example, it shows whether an extremal point is strictly inside a link ($0 < t < 1$). The respective linear program is then degenerate and, hence, the dual program has more than one solution.

Next we come over to the subdifferential of the distance function $\mathrm{d}(\mathbf{x}, C)$. To this end we represent $\mathrm{d}(\mathbf{x}, C)$ as the inf-convolution of two functions [45]:

$$\mathrm{d}(\mathbf{x}, C) = \inf_{\mathbf{y} \subset C} \|\mathbf{x} - \mathbf{y}\| = \inf_{\mathbf{y} \subset \mathcal{R}^n} \{\|\mathbf{x} - \mathbf{y}\| + \delta_C(\mathbf{y})\}$$
$$= (n \oplus \delta_C)(\mathbf{x}), \tag{4.15}$$

where the sign $\oplus$ implies an inf-convolution, which for functions $f(x)$ and $g(x)$ at a point x is

$$(f \oplus g)(x) = \inf_{\mathbf{y}} \{f(\mathbf{x} - \mathbf{y}) + g(y)\};$$

$\delta_C(\mathbf{x})$ is the indicator function of the convex set C [45]; and $n(\mathbf{x})$ is the norm of the vector $\mathbf{x}$.

Let a minimum of $\|\mathbf{x} - \mathbf{y}\|$ in (4.15) be attained at a point $\mathbf{y}_0 \subset C$, that is, $\mathbf{y}_0$ is the closest point of the polyhedron for $\mathbf{x}$. Then, using the expression of the subdifferential of two inf-convoluted functions written through the intersection of the subdifferentials of these functions [28], we get

$$\partial \mathrm{d}(\mathbf{x}, C) = \partial (n \oplus \delta_C)(\mathbf{x}) = \partial n(\mathbf{x} - \mathbf{y}_0) \cap \partial \delta_C(\mathbf{y}_0), \tag{4.16}$$

where $\partial \mathrm{d}(\mathbf{x}, C)$ is the subdifferential of the distance $\mathrm{d}(\mathbf{x}, C)$ at the point $\mathbf{x}$; $\partial n(\mathbf{x} - \mathbf{y}_0)$ the subdifferential of the norm at the point $\mathbf{x} - \mathbf{y}_0$; and $\partial \delta_C(\mathbf{y}_0)$ the subdifferential of the indicator function of the set C at the point $\mathbf{y}_0$. The subdifferential of the Chebyshev norm $\partial n(\mathbf{x})$ for $\mathbf{x} \neq 0$ is the convex hull of the vector set (see [45])

$$\mathbf{k}_j = (\mathrm{sgn} x_j)\mathbf{e}_j, \quad j \subset J_x,$$

where $\mathbf{e}_j$ are the unit vectors in $\mathcal{R}^n$, and J_x is the set of repeated indexes for which $|x_j| = n(\mathbf{x})$. Hence, $\partial n(\mathbf{x} - \mathbf{y}_0)$ is the convex polyhedral set which belongs to the polyhedral cone K_n whose generators are $\{\mathbf{k}_j\}$. In the circumstances, K_n is either an octant, or a quadrant, or a half-line spanned on these vectors.

The subdifferential $\partial\delta_C(\mathbf{y}_0)$ is the cone $N_C(\mathbf{y}_0)$ normal to the set C at the point $\mathbf{y}_0$ [45], i.e.,

$$\partial\delta_C(\mathbf{y}_0) = N_C(\mathbf{y}_0) = \{\mathbf{y}^*, \ \langle \mathbf{y}^*, \mathbf{z}-\mathbf{y}_0\rangle \leqslant 0, \ \forall \mathbf{z} \subset C\}.$$

In the particular case, $N_C(\mathbf{y}_0)$ is the polyhedral cone whose generators are the unit normals to the faces of polyhedron containing the point $\mathbf{y}_0$; speaking in other terms, the coordinates of the generator vectors are the coefficients in the first m basic constraints of the linear program in the form (4.14), which become the strict equalities at the extremal value of t. Thus, the vectors spanning on the subdifferential set $\partial d(\mathbf{x}, C)$ can be found as an intersection of the polyhedral convex cones $N_C(\mathbf{y}_0)$ and K_n, the former of which is specified by its generators and the latter can be defined by a system of linear homogeneous inequalities.

Stating this problem in the equivalent terms, the set of basic solutions to the initial homogeneous inequalities (the set of generator vectors of the cone $N_C(\mathbf{y}_0)$) is to be overcast by adding new inequalities describing the cone K_n. For the set $\partial d(\mathbf{x}, C)$ this results in the family of ND generator vectors $\mathbf{B}^j = [B_1^j, B_2^j, \ldots B_n^j]^T$, $j = \overline{1,ND}$, normalized in the norm of the adjoint space l_1, viz.,

$$\|\mathbf{B}^j\|_1 = \sum_{i=1}^{n} |B_i^j| = 1.$$

Thus $\partial d(\mathbf{x}, C)$ is a set of convex linear combinations of the form $\sum_{j=1}^{ND} \lambda_j \mathbf{B}^j$, where all $\lambda_j \geqslant 0$, and $\sum_{j=1}^{ND} \lambda_j = 1$.

As a possible approach to choosing the vector of subgradient, which is contained in the constraining inequality (4.9), it is suggested to evaluate the mean vector

$$\mathbf{B}^* = \frac{1}{ND} \sum_{j=1}^{ND} \mathbf{B}^j.$$

Note that some information on the subdifferential $\partial d(\mathbf{x}, C)$ can be gained from the variables of the dual linear program (4.14).

Analyzing the solutions of the dual program, we find the derivatives of the objective function (the function of distance $\partial d(\mathbf{x}, C)$) with respect to the directions corresponding to the right-hand sides of the last n constraints in (4.14), that is along the coordinate axes. Recalling that the derivative with respect to a direction y taken of a convex function $f(\mathbf{x})$ at a point x is the supporting function of the subdifferential $\partial f(\mathbf{x})$ as a function of y, we arrive at a parallelepiped (or a rectangle in the two-dimensional case) whose edges run parallel to the coordinate axes and the body contains the subdifferential set $\partial d(\mathbf{x}, C)$.

In the planar case ($n = 2$) it suffices to specify the subgradient $\nabla d(\mathbf{x}, C)$ so that it is directed along the diagonal of the rectangle that passes through the origin without resorting to the construction of $\partial d(\mathbf{x}, C)$ as an intersection of the sets, subject to (4.16).

4.4 SOFTWARE REALIZATION AND NUMERICAL MODELING

A software to compute the distances and their subdifferentials, along the lines of the algorithm considered in the previous

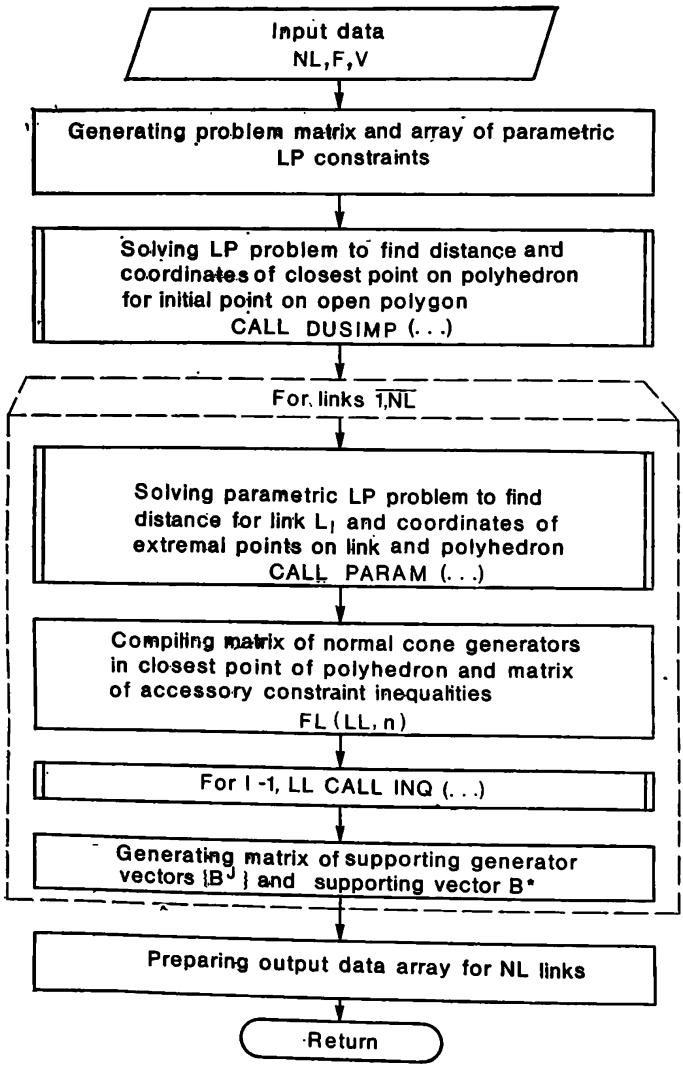

Input data
NL,F,V
Generating problem matrix and array of parametric LP constraints
Solving LP problem to find distance and coordinates of closest point on polyhedron for initial point on open polygon
CALL DUSIMP (. . .)
For links 1,NL
Solving parametric LP problem to find distance for link L_I and coordinates of extremal points on link and polyhedron
CALL PARAM (. . .)
Compiling matrix of normal cone generators in closest point of polyhedron and matrix of accessory constraint inequalities
FL (LL, n)
For I -1, LL CALL INQ (. . .)
Generating matrix of supporting generator vectors $\{B^J\}$ and supporting vector B*
Preparing output data array for NL links
Return

Fig. 4.1 Flow chart of the DIST subroutine

section, was written in FORTRAN IV as the program DIST which, in turn, calls subroutines DUSIMP, PARAM and INQ. The main steps of the program DIST are shown in Fig. 4.1. DUSIMP solves the LP problem (4.14) for the first link, for $t = 0$, by the dual simplex method with the resultant distance from the initial point $\mathbf{v}^1$ in the open polygon to C. The subroutine PARAM, employed in the cycle through all links, solves NL parametric linear programs of the type (4.14). The initial solution for the problem l at $t = 0$ ($l = \overline{2,NL}$) comes from the problem $l - 1$ in the form of the solution for $t = 1$.

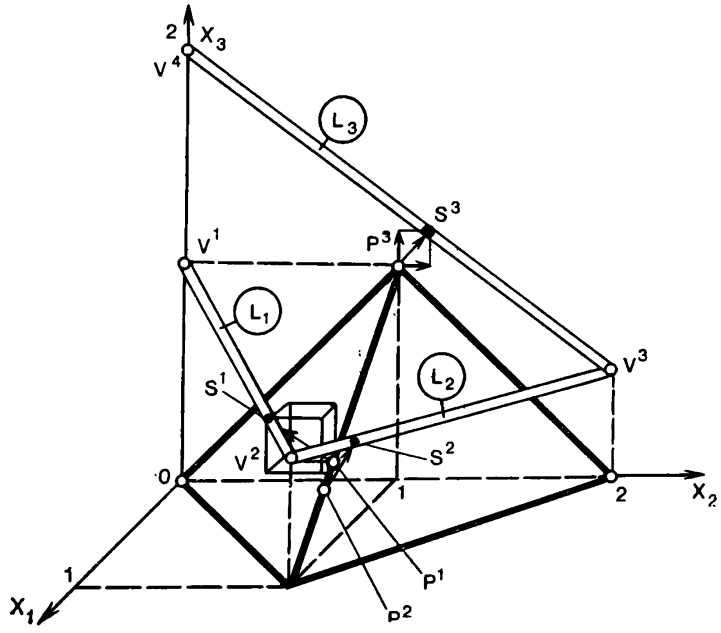


Fig. 4.2 Position of the links with respect to a tetrahedral body

To solve these linear programs with parametric constraint vectors the resort is made to a modification of the dual simplex method. The subroutines DUSIMP and PARAM request the subroutines ROW, COL and PIVCO; ROW chooses the pivotal row, COL chooses the pivotal column, and PIVCO transforms the condensed simplex table with the help of the Gaussian eliminating. The subroutine INQ constructs, for a given set of basic solutions to the original homogeneous inequalities, a new set corresponding to an extension of the original system of constraints for one more inequality.

To illustrate the procedure, consider a three-bar linkage and a tetrahedron C placed in space as shown in Fig. 4.2. We have:

$n = 3$, $NL = 3$, $m = 2$ (the other two faces are described by simple constraints $x \geqslant 0$ of the LP problem), and

$$[\mathbf{v}^1 \mid \mathbf{v}^2 \mid \mathbf{v}^3 \mid \mathbf{v}^4] = \begin{vmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 0 \\ 1 & 0.5 & 0.5 & 2 \end{vmatrix}, \quad \mathbf{F} = \begin{vmatrix} 1 & 1 & 1 & 2 \\ 1 & -1 & 1 & 0 \end{vmatrix}.$$

The results obtained by running the above algorithm on a computer are summarized in the table below. Figure 4.2 displays also the extremal points $\mathbf{S}^l$ on the links and the corresponding closest points $\mathbf{P}^l$ on the polyhedron. Note in passing that when an extremal point is not unique, the algorithm determines the point having the smallest t.

Table 4.1

		Link 1	Link 2	Link 3
Iteration steps		4	4	3
Extremal value of t^l		0.8	0.167	0.429
Minimal distance, z^l (t^l)		0.2	0.167	0.143
Coordinates of extremal point $\mathbf{S}^l$ on link L_l	x_1:	0.8	0.833	0.000
	x_2:	0.8	1.167	1.143
	x_3:	0.6	0.500	1.143
Coordinates of closest point $\mathbf{P}^l$ of polyhedron	x_1:	0.6	0.667	0.0
	x_2:	1.0	1.000	1.0
	x_3:	0.4	0.333	1.0
Coordinates of vectors $\mathbf{B}^j$ generating subdifferential of distance at point $\mathbf{S}^l$ ($\|\mathbf{B}^j\|_1 = 1$)	$\mathbf{B}^1$ x_1:	0.333	0.333	0.0
	$\mathbf{B}^1$ x_2:	-0.333	0.333	0.5
	$\mathbf{B}^1$ x_3:	0.333	0.333	0.5
	$\mathbf{B}^2$ x_1:	0.5	0.5	0.0
	$\mathbf{B}^2$ x_2:	0.0	0.0	0.0
	$\mathbf{B}^2$ x_3:	0.5	0.5	1.0
Coordinates of mean supporting vector $\mathbf{B}^*$	x_1:	0.417	0.417	0.00
	x_2:	-0.167	0.167	0.25
	x_3:	0.417	0.417	0.75

Figure 4.3 plots the distance from the links to the tetrahedron, along the whole open polygon $\mathbf{v}^1\mathbf{v}^2\mathbf{v}^3\mathbf{v}^4$, as a function of t. To demonstrate the geometry, the subdifferential $\partial d(\mathbf{x}, C)$ at point $\mathbf{S}^1$ for link 1 is depicted in Fig. 4.4. In the discussed example, the normal cone to the tetrahedron in the closest point $\mathbf{P}^1$, $N_C(\mathbf{P}^1)$, is generated by two vectors, $0\mathbf{A}_1$ and $0\mathbf{B}_1$. The subdifferential of the Chebyshev norm at point $\mathbf{S}^1$-$\mathbf{P}^1$ is the convex hull spanned on the set of unit coordinate

vectors $\mathbf{e}_1$, $\mathbf{e}_2$, and $\mathbf{e}_3$. The intersection of the sets $N_C(\mathbf{P}^1)$ and $\partial n\,(\mathbf{S}^1\text{-}\mathbf{P}^1)$ yields the sought subdifferential $\partial \mathrm{d}\,(\mathbf{S}^1, C)$ which geometrically is the triangle OCD.

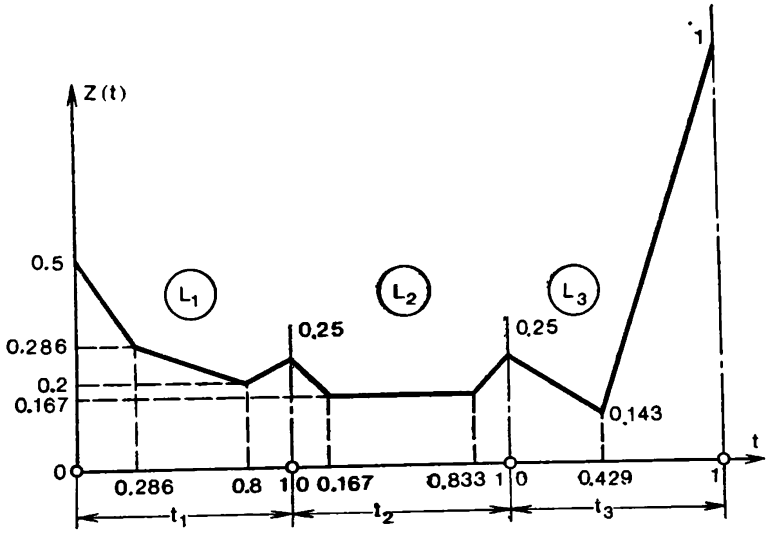


Fig. 4.3 Distance from the links to the tetrahedron

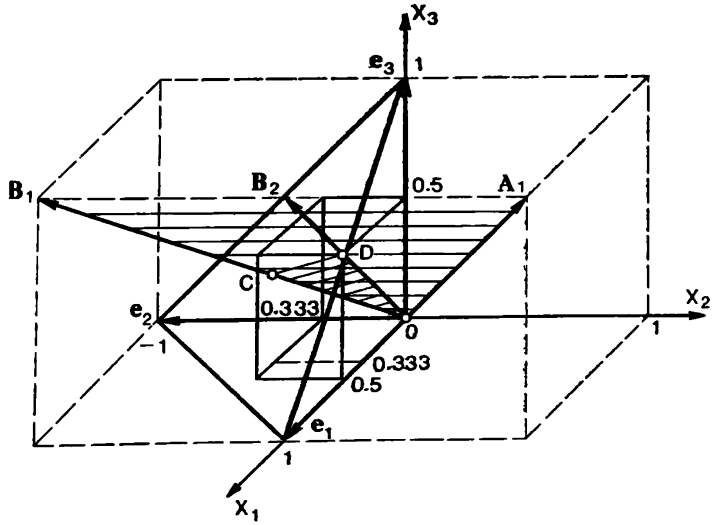


Fig. 4.4 Constructing the subdifferential of the distance at the point $\mathbf{S}^1$ on link 1

To the coordinates of the vertices of the parallelepiped circumscribed around the set OCD and having its faces parallel to the coordinate planes there correspond the derivatives with respect to the

directions of the coordinate axes, gained from the analysis of dual variables in the linear program (4.14).

Consider now, step by step, the computational scheme and the software of the path synthesis algorithm. As will be recalled, the linear algorithms of path synthesis and control are essentially based on the known relationship for small increments

$$\Delta \mathbf{S} = \mathbf{J}(\mathbf{q})\Delta \mathbf{q}, \tag{4.17}$$

where $\Delta \mathbf{S}$ is the command vector of small increments which the coordinates of the hand vector state $\mathbf{S}$ receive in an n-dimensional workspace. In the general case $\Delta \mathbf{S} = [\Delta \mathbf{x}^{\mathrm{T}} \vdots \Delta \mathbf{\Phi}^{\mathrm{T}}]^{\mathrm{T}}$ is a 6-tuple. Here, $\Delta \mathbf{x} = [\Delta x_1, \Delta x_2, \Delta x_3]^{\mathrm{T}}$ is the increment of the coordinates of the hand position vector $\mathbf{p}^N$; $\Delta \mathbf{\Phi} = [\Delta \Phi_1, \Delta \Phi_2, \Delta \Phi_3)]^{\mathrm{T}}$ is the vector of small increments in hand revolution around the axes of the inertial reference frame; $\Delta \mathbf{q} = [\Delta q_1, \Delta q_2, \ldots, \Delta q_N]^{\mathrm{T}}$ is the vector of small increments of the generalized coordinates; and $\mathbf{J}(q)$ is the $n \times N$ Jacobian matrix of the mapping $\mathbf{S} = f(\mathbf{q})$. The computational scheme of the path synthesis algorithm can be visualized as the following steps.

1. By a known configuration $\mathbf{q}[k]$, compute for the kth node on the path the matrix of the hand state $\mathbf{T}^N[k]$, the Jacobian matrix for the hand $\mathbf{J}[k]$, and the position vectors and sensitivity matrices for the vertices of the open polygon.

2. Compute the distances from the links to the obstacles. Determine the extremal points on the links and the subgradients of the distances and the Jacobian matrices for these points.

3. Evaluate the command vector $\Delta \mathbf{S}[k]$ at the kth node by the matrices $\mathbf{T}^N[k]$ and $\mathbf{T}_f^N$ as

$$\Delta \mathbf{S}[k] = \frac{h[k]}{\| \rho[k] \|_2} [[\rho[k]]^{\mathrm{T}} \vdots [\Phi[k]\, \mathbf{n}[k]]^{\mathrm{T}}]^{\mathrm{T}},$$

where $h[k]$ is a specified step size on moving along the path, which is linearly dependent upon the minimum distance for all links and varies between its maximum and minimum values, $h_{\max}$ and $h_{\min}$; $\rho[k] = \mathbf{p}^N[k] - \mathbf{p}_f^N$; $\| \rho[k] \|_2$ is the current distance of the hand to the state of destination; and $\Phi[k]$ and $\mathbf{n}[k]$ are, respectively, the angle and the unit vector defining the direction of a required rotation to move the hand from the current state $\mathbf{T}^N[k]$ to the terminal state $\mathbf{T}_f^N$. The values of $\Phi[k]$, and $\mathbf{n}[k]$ can be found [43] by the rotation matrix $\mathbf{B}(\Phi[k], \mathbf{n}[k]) = \mathbf{1}_f [\mathbf{l}[k]]^{\mathrm{T}}$.

4. Compute by the known command vector $\Delta \mathbf{S}[k]$ the vector of increments of the generalized coordinates $\Delta \mathbf{q}[k]$, subject to

$$\min_{\Delta \mathbf{q}[k]} \{ \sum_{i=1}^{N} \gamma_i |\Delta q_i[k]| + \max_{j=\overline{1,n}} |(\Delta \mathbf{S}[k] - \mathbf{J}[k]\Delta \mathbf{q}[k])_j| \}, \tag{4.18}$$

that is, such that minimizes the function composed of the weighted sum of the absolute values of increments of the generalized coordinates multiplied by the weighting coefficients γ_i plus the error in the Chebyshev norm sense of the solution to the system of equations (4.17) subject to the constraints (4.9), (4.10), and

$$q_{i,\min} - q_i[k] \leqslant \Delta q_i[k] \leqslant q_{i,\max} - q_i[k],\ i = \overline{1,N}. \tag{4.19}$$

Substituting $\Delta q_i[k] = x_i - x_{N+i}$, with both x_i and $x_{N+i} \geqslant 0$, and introducing an additional variable $x_{2N+1} \geqslant 0$ for the error in (4.18) reduces the above minimizing problem (4.18) subject to (4.9), (4.10) and (4.19) to the following ordinary linear program (see [5,15]):

$$\min \mathbf{c}^T \mathbf{x},\ \mathbf{A}\mathbf{x} \leqslant \mathbf{B},\ \mathbf{x} \geqslant 0, \tag{4.20}$$

where $\mathbf{x} = [x_1, x_2, \ldots, x_{2N+1}]^T$

$$\mathbf{c}^T = [\gamma^T \mid \gamma^T \mid 1],$$

$$\gamma^T = [\gamma_1, \gamma_2, \ldots, \gamma_N],$$

and $\mathbf{A}$ is the condition matrix of dimension $[2(n+N) + NL \times NC + NJ]$ by $(2N+1)$. Here, NC is the number of obstacles of the convex polyhedron type, and NJ the number of constraints of the type (4.10). $\mathbf{B}$ is seen to be the vector of the right-hand sides of the constraints.

Since the coefficients of the objective function are nonnegative it would be well-advised to solve this linear program by the dual simplex method [5,15]. Knowing both $\Delta\mathbf{q}[k]$ and $\mathbf{q}[k]$, determine the configuration for the $(k+1)$st node

$$\mathbf{q}[k+1] = \mathbf{q}[k] + \Delta\mathbf{q}[k]$$

and recur to Step 1.

The algorithm is deemed to perform normally if, completely run, it brings the hand in a given neighborhood of the destination state.

A computer realization of this algorithm is a program PLAN written in FORTRAN IV. It requests the following main subroutines: POSITN, DIST, DUSIMP, MATRIX, and STEP. Its flow diagram showing the hierarchy of the program units is presented in Fig. 4.5. Figures imply allocated memory in Kbytes. The assigned jobs are as follows. POSITN determines by a given configuration the position and orientation of the links and the Jacobian matrices for the link terminal points. MATRIX generates the condition matrix $\mathbf{A}$ and the vector of constraints $\mathbf{B}$ for the LP program in the form (4.20). DUSIMP solves this linear program by the dual simplex method. STEP issues the size of path synthesis steps as a function of minimal distance for all links. The subroutines NORMT and NORM1 requested by the subroutine DIST normalize vectors in the Chebyshev norm and in the norm of l_1 space, respectively.

To conclude, consider two examples.

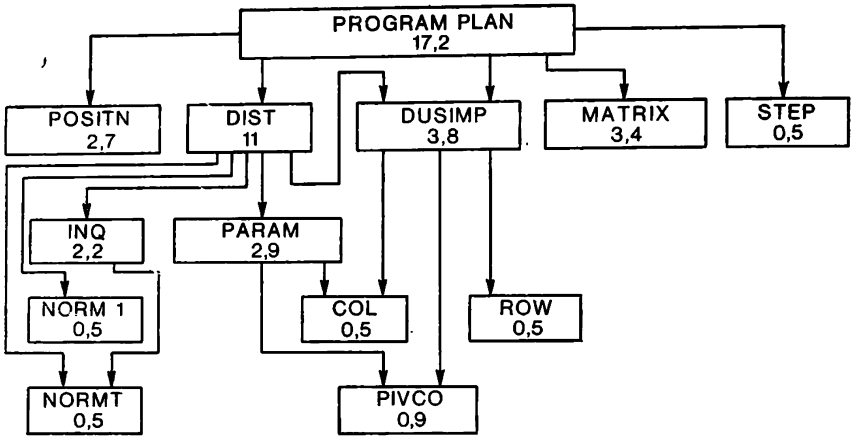


Fig. 4.5 Hierarchy of the path synthesis algorithm

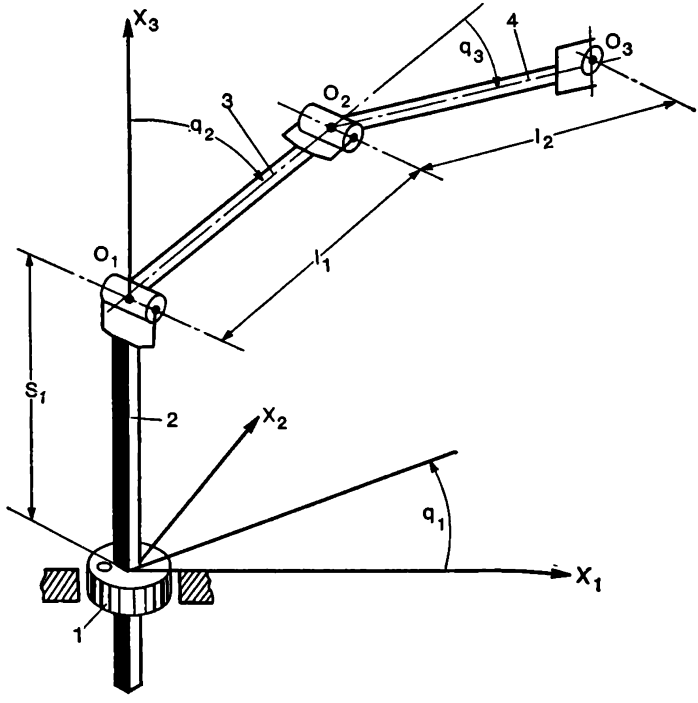


Fig. 4.6 Kinematic diagram of a 4-bar manipulator

Illustrative example 1. A 4-bar manipulator (three revolute and one prismatic joint, the kinematic diagram in Fig. 4.6) works over the

table plane X_1X_2 in 3-D space containing an obstacle of the rectangular parallelepiped type (Fig. 4.7*a*).

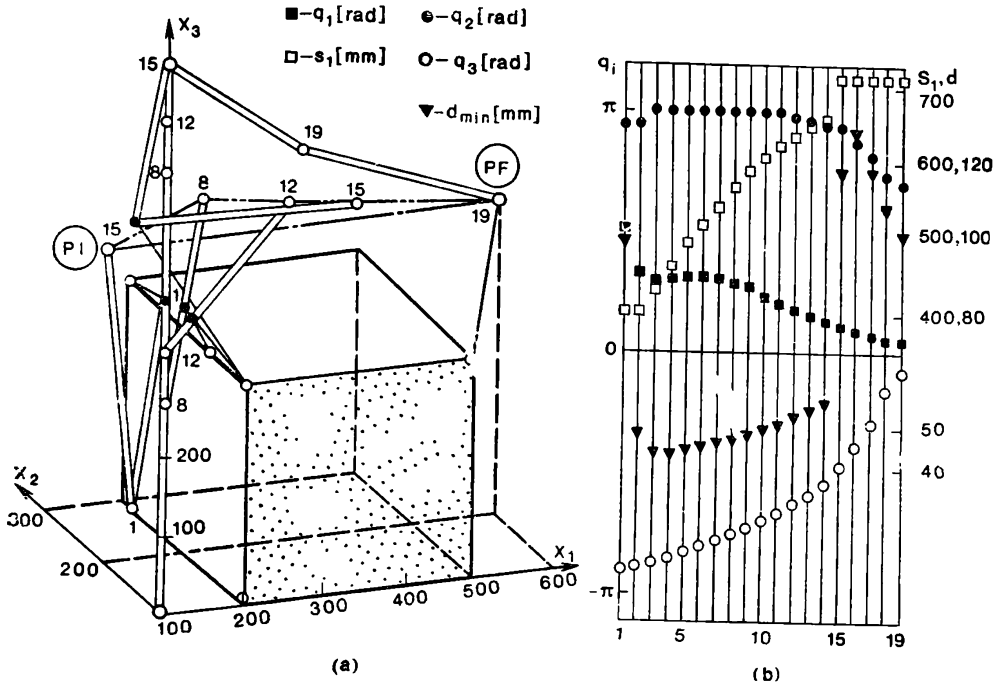


Fig. 4.7 Linkage configurations (a) and plots of generalized coordinates q_i and $d_{\min}$ (b) in moving around an obstacle in the form of a rectangular parallelepiped

The linkage parameters are: the links $l_1 = l_2 = 300$ mm; the constraints on displacements in the joints are $|q_i| \leqslant \pi$, $i = 1, 2, 3$; 100 mm $\leqslant S_1 \leqslant$ 1000 mm; the weighting coefficients of the objective function $\gamma_i = 0.1$, $i = \overline{1,4}$; the specified minimal and maximal step sizes along the path $h_{\min} = 30$ mm and $h_{\max} = 60$ mm; the minimum clearance to obstacles $r_{\min} = 50$ mm; the initial configuration of the linkage $\mathbf{q}_s = [1.5708, 400, 2.96, -2.8]^T$; the destination state of the hand $\mathbf{p}_f = [600, 200, 400]^T$.

The matrix $\mathbf{F}$ of the parallelepiped faces is

$$\mathbf{F} = \begin{vmatrix} 1 & 0 & 0 & 500 \\ -1 & 0 & 0 & -200 \\ 0 & 1 & 0 & 300 \\ 0 & -1 & 0 & -100 \\ 0 & 0 & 1 & 300 \end{vmatrix}$$

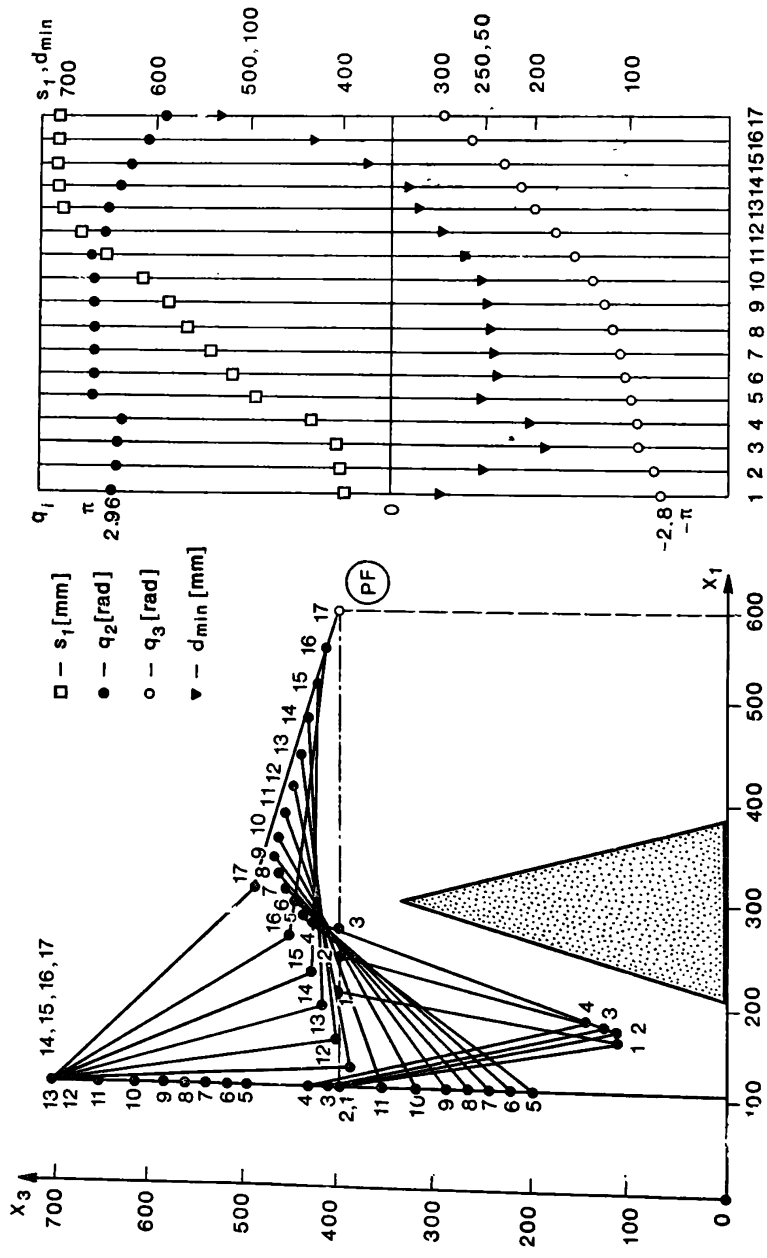


Fig. 4.8 Admissible configurations of the linkage (a) and variations of the generalized coordinates q_i and d_{min} (b) in planar moving around an isosceles triangle

The table plane as an obstacle has been considered by two additional constraints of the form (4.10) for the points O_2 and O_3 on the linkage. The computed number of nodal points on the path grid is $M = 19$.

Figure 4.7*a* demonstrates the configurations of the linkage in the initial, terminal and some intermediate points. The extremal points in these configurations are indicated both on the links and their counterparts on the parallelepiped. Figure 4.7*b* presents the plots showing how the generalized coordinates and the minimal distance to an obstacle, $d_{\min}$, vary for all the robot links at the nodal points of the path.

Illustrative example 2. The robot linkage of the previous example performs planar motions ($q_1 = 0$) avoiding an obstacle in the form of an isosceles triangle. The constructive constraints and parameters $\{\gamma_i\}$, $r_{\min}$, $h_{\min}$ and $h_{\max}$ are the same as in the previous example.

Figure 4.8*a* and *b* depicts the succession of the admissible configurations and plots the variations of the generalized coordinates and $d_{\min}$.

List of Symbols

$\mathbf{q}$	vector of the generalized coordinates (configuration of the robot linkage)
$\mathbf{J}_l$	Jacobian matrix of the mapping
$\mathbf{x}_l$	coordinate vector of a linkage point in space $\mathscr{R}^n$
$\mathrm{d}(\mathbf{x}, C) = \min_{y \subset C} \|\| \mathbf{x} - \mathbf{y} \|\|$	distance from $\mathbf{x}$ to the convex set C
$\partial \mathrm{d}(\mathbf{x}, l)$	subdifferential of the distance $\mathrm{d}(\mathbf{x}, C)$ at point $\mathbf{x}$

5 Modeling Robot Control in Assembly Operations

F. M. Kulakov, D. Sc. (Tech.)

5.1 GENERAL

The latest developments in robot engineering have made it fairly obvious that the field of robot application in industry could be drastically widened by endowing the robot with a capability to manipulate kinematically constrained bodies. The decisive breakthrough would apparently be feasible in a variety of assembly operations which so far have been performed manually in a majority of lines.

Two problems should be solved in improving the robot dexterity in handling constrained bodies. One consists in making the robot sense the stresses on its end effector, that is in measuring and reduction of the data on forces and moments exerted on the end effector in handling such bodies. The second problem is to generate a control of robot motions on the above data.

The first problem has been brought to a complete and efficient solution in a number of works. Reference [52], for example, suggests several versions of a sensor which is mounted in the manipulator wrist to measure the six components to the reaction vector $\mathbf{G} = [M_1, M_2, M_3, P_4, P_5, P_6]^T$; the first three components are the principal moments of the reaction forces exerted on the gripper and measured about the axes of a gripper-tied frame of reference; the other three are the projections to these axes of the principal reaction force vector.

The second problem appears to have been developed to a lesser extent. The body of this chapter will, therefore, be mainly focused on this aspect.

Physically, the considered interaction may be viewed as a seizure by the gripper of a body on which are imposed kinematic constraints. In assembling, such a body can be a bar in supports, a nut being screwed on a bolt, a lever, etc. We shall assume that the seizure is such that any slippage or turning in the gripper is impossible.

5.2 MATHEMATICAL MODEL

The equation of motion of the manipulator linkage accounting for the constraints has the form

$$\mathbf{A}(\mathbf{q})\ddot{\mathbf{q}} + \sum_{s=1}^{m} (\dot{\mathbf{q}}^T \mathbf{D}_s(\mathbf{q})\dot{\mathbf{q}})\mathbf{e}_s - \mathbf{Q}_w(\mathbf{q}) - \mathbf{R}(\mathbf{q})\mathbf{G} - \mathbf{Q}_d = \mathbf{Q}_a, \qquad (5.1)$$

where $\mathbf{q}$ = m-dimensional vector of the robot generalized coordinates;

$\mathbf{A}(\mathbf{q}) = [a_{ij}]$
and
$\mathbf{D}_s(\mathbf{q}) = [d^s_{ij}] = m \times m$ matrices whose elements are related by
$d^s_{ij} = \frac{1}{2}\left(\frac{\partial a_{is}}{\partial q_t} + \frac{\partial a_{ts}}{\partial q_i} - \frac{\partial a_{it}}{\partial q_s}\right)$;
$\mathbf{Q}_w(\mathbf{q}) = m$-dimensional vector of forces defined by the weights of the robot links and base;
$\mathbf{R}(\mathbf{q}) = m \times 6$ matrix defined by the kinematic structure of the arm linkage;
$\mathbf{Q}_a = m$-dimensional vector of moments and forces due to actuators;
$\mathbf{Q}_d = m$-dimensional vector of forces and moments due to disturbing forces.

The equation of motion for the linkage with the respective actuators (electric motors in this case) can be written as

$$[\mathbf{A}(\mathbf{q}) + \bar{\mathbf{J}}_a]\ddot{\mathbf{q}} + \sum_{s=1}^{m} [\dot{\mathbf{q}}^T \mathbf{D}_s(\mathbf{q})\dot{\mathbf{q}}]\mathbf{e}_s + [\mathbf{K}_{vf} + \mathbf{K}_M\mathbf{R}^{-1}]\dot{\mathbf{q}}$$
$$+ \mathbf{M}_{fr}\operatorname{sgn}(\dot{\mathbf{q}}) - \mathbf{Q}_w(\mathbf{q}) - \mathbf{R}(\mathbf{q})\mathbf{G} - \mathbf{Q}_d = \mathbf{K}_M\mathbf{R}^{-1}\mathbf{u}, \tag{5.2}$$

where $\mathbf{u} = m$-dimensional vector of controls (voltages across the armatures of actuator driving motors);
$\bar{\mathbf{J}}_a = m \times m$ diagonal matrix of the moments of inertia of the drives referred to the load;
$\mathbf{K}_M\mathbf{R}^{-1} = m \times m$ diagonal matrix in which each diagonal element is a ratio of a coefficient K_{Mi} giving a proportion between the driving moment and the current in the ith motor to the resistance R_i of its armature;
$\mathbf{K}_{vf} = m \times m$ diagonal matrix of the coefficients of viscous friction;
$\mathbf{M}_{fr}(\mathbf{q}) = [M_{1,fr}\operatorname{sgn}(\dot{q}_1), \ldots, M_{m,fr}\operatorname{sgn}(\dot{q}_m)]^T$ is the vector of moments due to friction forces referred to the load shaft.

We shall assume that the kinematic structure of the manipulator and all the parameters of the robot and actuators are known, whereas the nature of the constraints cannot be practically specified in advance. Hence, the term $\mathbf{R}(\mathbf{q})\mathbf{G}$ on the right-hand side of (5.1) and (5.2) is unknown because it is defined by these constraints. Its value, though, can be determined provided that we know the vector $\mathbf{G}$ of the reaction forces applied to the gripper by a seized body. This vector can be measured, say, with a six-component sensor mentioned above.

With the object of control so defined we may state the problem of control as follows. It is required to synthesize the system of control which will generate a vector of actuator control $\mathbf{u}$ such that will

guide the object of control, irrespective of possible disturbances and independently of the nature of constraints imposed, from a known initial state to a target state specified by the human operator. The initial state is described by the vector of the generalized phase coordinates $\mathbf{q}_{ph} = [\dot{q}, q]^T$ at $t = 0$, or the respective vector of the phase coordinates $\mathbf{x}_{ph} = [\dot{x}, x]^T$, and the target state is known to belong to the set of all possible target states, $\mathbf{x}_{ph} \subset \mathbf{X}_t$. The relation between the vector of position of the gripper in **x** space and the vector of the generalized coordinates **q** is known provided that the structural scheme of the robot is settled.

A translation of an object to a target state should be performed subject to the constructional limitations which require that the magnitude of the reaction forces applied to the gripper must not go above some allowable value

$$| \mathbf{G} | \leqslant G_{cr}, \tag{5.3}$$

otherwise the arm can be broken down or the translated body can be damaged or jammed.

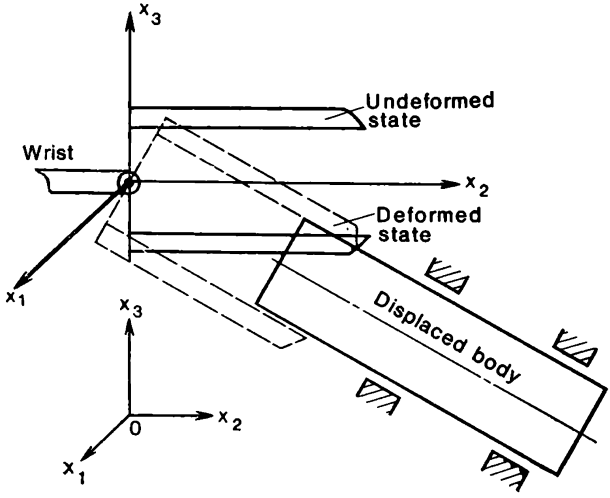


Fig. 5.1 Kinematic diagram of the gripper handling a constrained body

We shall assume that in our robot the gripper is coupled with the wrist flexibly. Then the vector **G** can be represented by

$$\mathbf{G} - \mathbf{C}\Delta\mathbf{y} \text{ with } |\Delta\mathbf{y}| \leqslant \varepsilon, \tag{5.4}$$

where $\mathbf{C} = 6 \times 6$ flexibility matrix;

$\Delta \mathbf{y} = [\theta_1, \theta_2, \theta_3, \Delta z_1, \Delta z_2, \Delta z_3]^T$, a six-dimensional vector describing the small deviations of the gripper from the "natural", undeformed state (Fig. 5.1) in which the gripper would be, should all the constraints be lifted; and

ε = small quantity.

The variables θ_i and Δz_i ($i = 1, 2, 3$) describe, with respect to the gripper-tied reference frame, the increments of small rotation and sliding displacement of the characteristic point, respectively, as the gripper deviates from the "natural" state.

The vector of deviations $\Delta \mathbf{y}$ can be expressed through the difference in position vectors of deformed, $\mathbf{x}_{def}$, and undeformed, $\mathbf{x}$, states as follows:

$$\Delta \mathbf{y} = \mathbf{B}(\psi, \nu, \varphi)[\mathbf{x}_{def} - \mathbf{x}], \tag{5.5}$$

where $\mathbf{x} = [\psi, \nu, \varphi, x_{n1}, x_{n2}, x_{n3}]^T$ contains the Euler angles describing the rotation of the gripper and the coordinates of its displacement with respect to the inertial frame of reference,

$$\mathbf{B} = \left| \begin{array}{c|c} \mathbf{W}(\psi, \nu, \varphi) & 0 \\ \hline 0 & \alpha(\psi, \nu, \varphi) \end{array} \right|,$$

$$\mathbf{W}(\psi, \nu, \varphi) = \begin{vmatrix} \sin\nu\sin\varphi & \cos\varphi & 0 \\ \sin\nu\cos\varphi & -\sin\varphi & 0 \\ \cos\nu & 0 & 1 \end{vmatrix},$$

and

$$\alpha(\psi, \nu, \varphi) = \left| \begin{array}{ccc} \cos\varphi\cos\psi - \sin\varphi\sin\psi\cos\nu & & \\ -\sin\varphi\cos\psi - \cos\varphi\sin\psi\cos\nu & & \\ \sin\nu\sin\psi & & \\ \hline \cos\varphi\sin\psi - \sin\varphi\cos\psi\cos\nu & \sin\varphi\sin\nu & \\ -\sin\varphi\sin\psi + \cos\varphi\cos\psi\cos\nu & \cos\varphi\sin\nu & \\ -\sin\nu\cos\psi & \cos\nu & \end{array} \right|.$$

If the nature of constraints is known in advance, then $\mathbf{x}_{def}$ is a known function of $\mathbf{x}$, that is, $\mathbf{x}_{def} = \mathbf{x}_{def}(\mathbf{x})$ and the constraint (5.3) can be written as

$$|\mathbf{G}| = |\mathbf{C}\Delta\mathbf{x}| = |\mathbf{CB}(\psi, \nu, \varphi)[\mathbf{x}_{def}(\mathbf{x}) - \mathbf{x}]| < G_{cr}. \tag{5.6}$$

This constraint defines the domain of the allowable values for the position vector $\mathbf{x}$.

When the kinematics of the robot under consideration is known, the components of the vector $\mathbf{x}$ in (5.5) can be expressed through the current value of the vector of generalized coordinates, employing to this end the kinematic relationship $\mathbf{x} = \mathbf{x}(\mathbf{q})$. The vector $\mathbf{x}_{def}$ can,

in principle, be also expressed via the vector $\mathbf{q}$. Hence, the constraint (5.3) in the generalized coordinates will be

$$|\mathbf{G}| = |\mathbf{C}\Delta\mathbf{y}| = |\mathbf{CB}(\psi(\mathbf{q}), \nu(\mathbf{q}), \varphi(\mathbf{q}))[\mathbf{x}_{def}(\mathbf{q}) - \mathbf{x}(\mathbf{q})]| < G_{cr}. \quad (5.7)$$

In this form the constraint defines the domain of the allowable values for the vector of generalized coordinates. Note, however, that if the nature of the constraints is not specified in advance, then the dependence $\mathbf{x}_{def}(\mathbf{q})$ in (5.7) is not known, neither is the dependence of $\mathbf{x}_{def}$ on $\mathbf{x}$ in (5.6).

Therefore, generally speaking, the dependence of the above constraints on $\mathbf{x}$ and $\mathbf{q}$ is not defined. The dependencies $\mathbf{x}_{def} = \mathbf{x}_{def}(\mathbf{q})$ and $\mathbf{x}_{def} = \mathbf{x}_{def}(\mathbf{x})$, however, can always be chosen such that the constraints (5.6) and (5.7) become more severe than stated. For example, we can fix $\mathbf{x}_{def}$ in (5.6) and (5.7) in the form

$$\mathbf{x}_{def} = \mathbf{B}^{-1}(\psi, \nu, \varphi)\mathbf{C}^{-1}\mathbf{G}_m + \mathbf{x}_m = \text{constant}, \quad (5.8)$$

where $\mathbf{G}_m$ is the measured magnitude of the reaction force vector for which the gripper assumes the position $\mathbf{x}_{def}$; the gripper would take the position $\mathbf{x}_m$ if all the constraints were lifted at the moment when $\mathbf{G} = \mathbf{G}_m$. A predicted variation of the reaction force vector will then be

$$\mathbf{G} = \mathbf{G}_m + \mathbf{CB}(\psi, \nu, \varphi)[\mathbf{x}_m - \mathbf{x}]. \quad (5.9)$$

The constraints placed on the domain of vector $\mathbf{x}$ will then be written as

$$|\mathbf{G}_m + \mathbf{CB}(\psi, \nu, \varphi)[\mathbf{x}_m - \mathbf{x}]| < G_{cr}, \quad (5.10)$$

and on the domain of vector $\mathbf{q}$ as

$$|\mathbf{G}_m + \mathbf{CB}(\psi(\mathbf{q}_m), \nu(\mathbf{q}_m), \varphi(\mathbf{q}_m))[\mathbf{x}_m(\mathbf{q}_m) - \mathbf{x}(\mathbf{q})]| < G_{cr}. \quad (5.11)$$

Obviously, the function $\mathbf{x}_m = \mathbf{x}(\mathbf{q}_m)$ is uniquely defined by the vector $\mathbf{q}_m$ measured at the moment when $\mathbf{G} = \mathbf{G}_m$.

With this selection of $\mathbf{x}_{def}$ the constraints (5.8) and (5.9) are more stringent than they would be in a practical situation with any nature of constraints, since they in fact do not allow for a displacement subject to the constraints on the seized body. This, however, can only diminish the value of the reaction force modulus in comparison with the case when $\mathbf{x}_{def}$ is fixed. We prove thus the above assertion stating that the constraints are most severe at $\mathbf{x}_{def} = \text{constant}$. They hold absolutely when the body is constrained so that any movement is excluded. With all other modes of constraining, the allowable domain of $\mathbf{x}$ or $\mathbf{q}$ always includes the set defined by (5.10) and (5.11) so that a path of controlled body or a portion of the path which satisfy these constraints will always fall within the actual constraints.

In place of the constraints in the form (5.3), we can use the constraints built subject to the requirement that the modulus of the term $\mathbf{R}(\mathbf{q})\,\mathbf{G} = \mathbf{Q}_R$ appearing due to the reaction forces in the sum of the vector of the generalized forces must not exceed a given critical value, that is,

$$|\mathbf{R}(\mathbf{q})\,\mathbf{G}| = |\mathbf{Q}_R| \leqslant Q_{R,\mathrm{cr}}. \tag{5.12}$$

This constraint can be represented as a function of $\mathbf{q}$ if we substitute for $\mathbf{G}$ in (5.12) the expression

$$\mathbf{G} = \mathbf{CB}(\psi(\mathbf{q}), \nu(\mathbf{q}), \varphi(\mathbf{q}))\,[\mathbf{x}_{\mathrm{def}}(\mathbf{q}) - \mathbf{x}(\mathbf{q})]. \tag{5.13}$$

or

$$\mathbf{G} = \mathbf{CR}^{\mathrm{T}}(\mathbf{q})\,[\mathbf{q}_{\mathrm{def}} - \mathbf{q}], \tag{5.13a}$$

where $\mathbf{q}_{\mathrm{def}}$ and $\mathbf{q}$ stand, respectively, for the deformed and current state of the gripper. This follows from (5.4) subject to $\Delta\mathbf{y} = \mathbf{R}^{\mathrm{T}}(\mathbf{q})\,\Delta\mathbf{q} = \mathbf{R}^{\mathrm{T}}(\mathbf{q})\,[\mathbf{q}_{\mathrm{def}} - \mathbf{q}]$.

5.3 POSSIBLE APPROACHES TO PROBLEM OF CONTROL

We may adopt one of two ways in synthesizing the control system that solves the problem of control stated in the previous section.

On the lines of the first approach, the problem of control is further subdivided, conditionally, into two simpler ones, as is frequently made in handling complex problems of optimal control. In the first part, we are to synthesize the corresponding programmed path of object motion as a law describing how the vector of generalized coordinates changes in time, $\mathbf{q}_{\mathrm{pr}} = \mathbf{q}_{\mathrm{pr}}(t)$, in response to the statement. In addition to this law, we are to devise the program control $\mathbf{u}_{\mathrm{pr}} = \mathbf{u}_{\mathrm{pr}}(t)$ which will guide the controlled body along the programmed path in the absence of disturbances.

In the second part, we are to generate the correcting component of the control as a function $\mathbf{u}_{\mathrm{st}} = \mathbf{u}_{\mathrm{st}}(\Delta\mathbf{q}, \dot{\mathbf{q}})$ which is to force the object to resume the programmed path from which it deviated under the influence of disturbances. The function $\mathbf{u}_{\mathrm{st}}$ is designed to make the motion stable and is effected by deviating the vector of generalized phase coordinates of its programmed values.

As can be readily ascertained, in the framework of this approach, the problem constraints are satisfied by solving the first of the mentioned subproblems, viz., the programmed path and control are constructed so that the reaction forces acting on the end effector would not exceed the critical boundaries, that is, satisfy (5.3).

With the second approach, extensively treated in [32] the problem of control is also partitioned into two subproblems. The first

part consists in generating the path of object motion from an initial to a target state as a function of time $\mathbf{q} = \mathbf{q}(t)$, yet, contrary to the previous case, with no allowance for the reaction forces acting on the end effector. It also involved devising the program control that will guide the object along the synthesized path in the absence of mentioned constraints and disturbances.

The second subproblem is to generate the correcting component of the control vector $\mathbf{u} = \mathbf{u}(\Delta\dot{\mathbf{q}}, \Delta\mathbf{q}, \mathbf{G})$ as function of deviations of the generalized phase coordinates from the program values and the reaction forces acting on the end effector. This component of control, firstly, corrects the path yielded by solving the first subproblem so that the forces of reaction could not exceed the critical values and, secondly, compensates, accurate to within small values, the deviations from the path caused by disturbances.

5.4 MINIMIZING OBJECTIVE FUNCTIONAL OVER POSITION VECTOR DOMAIN

Consider now several methods suggested to solve the control problem by way of the first approach. They differ mainly by the algorithms of program path synthesis, that is, by how the first subproblem is solved.

As far as the second subproblem is concerned, the experience gained in the development of master-slave manipulator systems indicates that it can be solved by formulating the control in the form

$$\mathbf{u}_{st} = \mathbf{K}_1 (\dot{\mathbf{q}}_{pr}(t) - \dot{\mathbf{q}}) + \mathbf{K}_2 (\mathbf{q}_{pr}(t) - \mathbf{q}), \tag{5.14}$$

where $\dot{\mathbf{q}}_{pr}(t)$, $\mathbf{q}_{pr}(t)$, and $\mathbf{q}(t)$, $\dot{\mathbf{q}}(t)$ are, respectively, programmed and measured variations in time of the generalized coordinate vectors and their derivatives; $\mathbf{K}_1$ and $\mathbf{K}_2$ are the square matrices of appropriately selected coefficients of velocity and position.

Turning back to the first approach, a first solution technique suggests to synthesize a program path and control by a step-by-step algorithm, that is, by a sequence of hierarchically connected procedures. The first procedure is to generate the position vector, say ith, $\mathbf{x} = \mathbf{x}^i$ on the path to target state such that, firstly, is closer to the target than the previous vector $\mathbf{x}^{i-1}$, computed by the same procedure on the previous step, and, secondly, has the reaction forces due to constraints not above the specified critical value, i.e. $|\mathbf{G}| \leqslant G_{cr}$.

Implementing the technique of numerical analysis, this procedure may be reduced to the construction of a minimizing sequence of the arguments of some functional which depends on vector $\mathbf{x}$ and is a minimum at $\mathbf{x} \subset \mathbf{X}_t$.

When the set of the target states consists of a sole element $\mathbf{x}_t$, then such a functional may be written as

$$\epsilon(\mathbf{x}) = |\mathbf{x}_t - \mathbf{x}|^2, \tag{5.15}$$

that is, as a squared distance between the two points of the Euclidean space to which correspond the vectors $\mathbf{x}_t$ and $\mathbf{x}$. We shall label it an error functional.

An algorithm to build the minimizing sequence should be chosen so that each of the sequence elements should be derived in a feasible area of the argument, bounded by the constraints on reaction forces (5.8). For this purpose one may employ the recursive techniques that would compute each next vector of undeformed state $\mathbf{x}^i$ by its previous value, moving in a feasible direction, namely,

$$\mathbf{x}^i = \mathbf{x}^{i-1} + \Delta\mathbf{x}^i,$$

where $\Delta\mathbf{x}^i$ is the increment computed subject to the constraint (5.8) with $\mathbf{x}^{i-1}$ substituted for $\mathbf{x}_m$.

The following procedure is to find the vector of generalized coordinates $\mathbf{q}^i$ for the respective position vector $\mathbf{x}^i$, found in the previous procedure. The computation consists in solving the system of transcendental equations $\mathbf{x} = \mathbf{x}(\mathbf{q})$ analitically or approximately. These equations are supplied by the known structure of robot linkage.

The third procedure is an interpolation. It is to form a program law to control the variation of the vector of generalized coordinates in the form of a polynomial function, linear in the simplest case, that is, to construct a segment of the program path as a function $\mathbf{q}_{pr} = \mathbf{q}_{pr}(t)$ from $\mathbf{q} = \mathbf{q}^{i-1}$, corresponding to $\mathbf{x} = \mathbf{x}^{i-1}$, to $\mathbf{q} = \mathbf{q}^i$. The necessary initial data are taken from the previous procedure as the computed $\mathbf{q}^{i-1}$ and $\mathbf{q}^i$ and their derivatives, and augmented by the boundary values for the derivatives of the generalized coordinates.

If the first and second derivatives at the ends of the segment are given equal to zero, which is a practically feasible case, then with the generalized coordinates equated at the ends to the computed $\mathbf{q}^{i-1}$ and $\mathbf{q}^i$ we get six boundary conditions.

In the formulated conditions, the sought law can be readily shown to be described for each of the generalized coordinates q_j by a polynomial of degree 5 in the form

$$q_j(t) = q_j^{i-1} + (q_j^i - q_j^{i-1})(10t^3/T_i^3 - 15t^4/T_i^4 + 6t^5/T_i^5), \tag{5.16}$$

where T_i is the time elapsed when the vector $\mathbf{q}$ varies from $\mathbf{q}^{i-1}$ to $\mathbf{q}^i$, being chosen subject to the limiting conditions imposed on the derivatives of the generalized coordinates [22].

With a sufficiently high-speed computer, it would be advisable to implement also the fourth procedure of finding the program control $\mathbf{u}_{pr} = \mathbf{u}_{pr}(t)$ guiding the body along the program path in the absence of disturbances. It converts essentially to calculating the left-hand side of the equation of motion (5.2) in which the vector $\mathbf{q}$ is replaced by its program value derived in the previous procedure, and the vectors $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ by the respective derivatives of $\mathbf{q}_{pr}(t)$. To compute the term $\mathbf{R}(\mathbf{q})\mathbf{G}$, its quantity $\mathbf{G}$ is replaced by the expression (5.9) where $\mathbf{x}_m$ and $\mathbf{G}_m$ are deemed constant.

These quantities, however, are constant with only the type of constraint that rules out any motions of the gripper; otherwise the computed $\mathbf{u}_{pr}$ will be wrong. Therefore, it would be advisable not to compute these terms in determining $\mathbf{u}_{pr}$, rather, to compensate the deviations from the programmed path, caused by these terms being left out and disturbances, by introducing the stabilizing component in the vector of control, $\mathbf{u}_{st} = \mathbf{u}_{st}(\mathbf{q}, \dot{\mathbf{q}})$.

Having performed all the above procedures, we get the programmed path $\mathbf{q}_{pr} = \mathbf{q}_{pr}(t)$ over the time interval $[t^{i-1}, t^i] = [t^{i-1}, (t^{i-1} + T_i)]$ which corresponds to changing $\mathbf{q}$ from $\mathbf{q}^{i-1}$ to $\mathbf{q}^i$, and the program control on this segment. The derived values, being converted to an analog form, are applied to the inputs of the robot servos to be realized as its motion over this time interval.

It should be noted that the realization of each ith step over the time interval $[t^{i-1}, t^i]$ calls for the constraints on the argument $\mathbf{x}$ of the functional (5.10) to be defined. This requires to measure the vectors $\mathbf{G} = \mathbf{G}_m^{i-1}$ and $\mathbf{q} = \mathbf{q}_m^{i-1}$ at the time $t = t^{i-1}$ and then, having $\mathbf{q}_m^{i-1}$ at hand, to determine the components in $\mathbf{x}_m$.

Obviously, running all the four procedures to generate $\mathbf{q}_{pr}$ and $\mathbf{u}_{pr}$ on a computer will be fairly time consuming for each segment of the path. Since a motion can be started only after the computational run for $\mathbf{q}_{pr}$ and $\mathbf{u}_{pr}$ for the segment is over, the gripper will, naturally, move in an interruptional mode. Hence, the third procedure will generate $\mathbf{q}(t)$ at each segment so that the first and second derivatives of $\mathbf{q}$ to be zero at the end points of the segment, i.e., $\dot{\mathbf{q}}^i = \ddot{\mathbf{q}}^i = \dot{\mathbf{q}}^{i-1} = \ddot{\mathbf{q}}^{i-1} = \mathbf{0}$. The variation of each generalized coordinate will thus be in accord with (5.16).

In order that the motion might proceed smoothly the computation of $\mathbf{q}_{pr}$ and $\mathbf{u}_{pr}$ for each next segment should be performed during the motion on the previous. A possible way to this goal is to modify the first procedure of the algorithm. The procedure in the original form computes $\mathbf{x}^i$ by $\mathbf{x}^{i-1}$, from the previous step, plus the increment $\Delta\mathbf{x}^i$ computed in the ith step of the algorithm: $\mathbf{x}^i = \mathbf{x}^{i-1} + \Delta\mathbf{x}^i$. In the modified form, it predicts the next-plus-one rather than the next

value of the vector, $\mathbf{x}^{i+1} = \mathbf{x}^{i-1} + \Delta\mathbf{x}^i$. The value of $\mathbf{x} = \mathbf{x}^i$ is generated on the $(i-1)$st step as $\mathbf{x}^i = \mathbf{x}^{i-2} + \Delta\mathbf{x}^{i-1}$, where the increment $\Delta\mathbf{x}^{i-1}$ is computed on the $(i-1)$st step of the algorithm. The condition $\varepsilon(\mathbf{x}^{i+1}) < \varepsilon(\mathbf{x}^i)$ and the constraint (5.10) must be satisfied, with $\mathbf{x}^{i-2}$ being substituted for $\mathbf{x}_m$ when checking for $\mathbf{x}^i$.

Thus, by the moment the gripper arrives at $\mathbf{x}^{i-1}$, i.e., passes between the end of the $(i-1)$st and the start of the ith segment, we can compute by the first procedure the value of $\mathbf{x}^{i+1}$ which belongs to the end of the ith and the start of the $(i+1)$st segment. Since $\mathbf{x}^i$ and $\mathbf{q}^i$ have been already obtained, we may realize the third procedure of interpolation and the fourth procedure of control generation for the $(i+1)$st segment, that is the control from $\mathbf{q}^i$ to $\mathbf{q}^{i+1}$, no longer waiting for the gripper to arrive at $\mathbf{x}^i$.

This algorithm enables the computation for the $(i+1)$st segment to be performed simulataneously with the motion of robot along the ith segment. Obviously, the motion will be smooth if the time required to realize all the four procedures for the $(i+1)$st segment is less than the time of motion on the ith segment. It may, though, impair the quality of the program being formed.

5.5 MINIMIZING OBJECTIVE FUNCTIONAL OVER GENERALIZED COORDINATE VECTOR DOMAIN

There exists another method of program path synthesis, which differs from that we discussed in the previous section in that it forms a current vector of generalized coordinates in one step rather than in two, as the previous method does. The procedure employed minimizes the error functional over the domain of vectors of generalized coordinates, rather than over that of position vectors $\mathbf{x}$.

To implement this procedure we need to change the variable in the functional (5.15) for $\mathbf{q}$, eliminating the variable $\mathbf{x}$ between the functional and the relation $\mathbf{x} = \mathbf{x}(\mathbf{q})$, which is known whenever the structure of the linkage is specified. The constraints on the variable should be also changed for the form (5.11) which is a function of vector $\mathbf{q}$.

Instead of using the functional in the form (5.15) we may employ more general and more computationally advantageous form of its representation, in which a minimum corresponds to a set of argument values (position vectors) rather than to a single vector. An example of such a representation is the functional

$$\varepsilon(\mathbf{w}) = |\mathbf{F}_t(\mathbf{w}) - \mathbf{w}|^2. \tag{5.17}$$

Similar to the previous case this functional is the squared difference between the vector of a target state position $\mathbf{F}_t(\mathbf{w})$ and a cur-

rent position vector w. However, this time the current position vector has the form

$$\mathbf{w} = [(\mathbf{x}_e)^1, \quad (\mathbf{x}_e)^t]^T.$$

It includes as components some points of the environment $(\mathbf{x}_e)^i$ represented in the coordinate system of the end effector. The vector of target position is defined as

$$\mathbf{F}_t(\mathbf{w}) = [\mathbf{F}^1, \ \mathbf{F}^2, \quad \mathbf{F}^t]^T,$$

where the element $\mathbf{F}^t = (F_1^t, F_2^t, F_3^t)$ is the three-dimensional vector whose components are defined as follows

$$\mathbf{F}_j^t = \begin{cases} b_j & \text{at } (x_{j,e})^i > b_j \\ a_j & \text{at } (x_{j,e})^i < a_j \\ (x_{j,e})^i & \text{at } a_j < (x_{j,e})^i < b_i, \end{cases}$$

$j = 1, 2, 3$; $i = 1, 2, \ldots, t$.

Obviously, this functional has a global minimum wherever all t points of the represented environment appear within a certain right parallelepiped, tied with the end effector (Fig. 5.2). The characteristic point of the gripper $x_0 = (x_{10}, x_{20}, x_{30})$ is connected with the

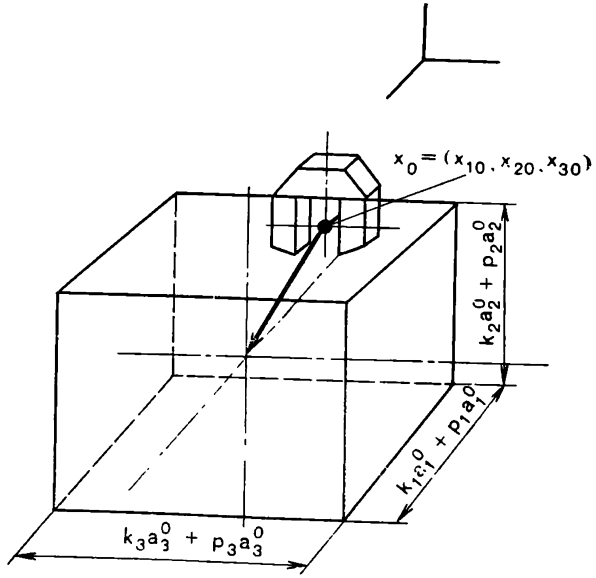


Fig. 5.2 Searching for the absolute minimum of an objective functional

parallelepiped by the vector $\mathbf{g} = (g_1, g_2, g_3)$. The parallelepiped has the dimensions $k_j a_j^0 + p_j a_j^0$; here a_j^0 $(j = 1, 2, 3)$ are the dimensions of the gripper workspace, and k_j and p_j are the deformation coefficients of the workspace.

When $a_j = -b_j$, $\mathbf{g} = \mathbf{0}$, and $k_i = p_i$, the shown target parallelepiped is symmetric with respect to the workspace of the gripper. The parallelepiped can be contracted into a single point coinciding with the characteristic point of the gripper (between its fingers) by letting

$$t = 1, \quad k_i = p_i = 0, \quad \text{and} \quad \mathbf{g} = \mathbf{0}.$$

It should be noted that to specify the functional for a certain operation of motion of a constrained body, it will suffice for the operator to introduce into the computer the coordinates of one or a few points of the environment as close as practicable to the contemplated path of object motion. It can be readily made by means of a location finder considered in [18].

The procedure of generating the program law $\mathbf{q}_{pr} = \mathbf{q}_{pr}(t)$ and program control $\mathbf{u}_{pr} = \mathbf{u}_{pr}(t)$ with this method is similar to that of the previous one. The minimization procedure forms the current vector of generalized coordinates $\mathbf{q}^i$. This vector along with the previous value $\mathbf{q}^{i-1}$ is used to construct $\mathbf{q}_{pr}$ for the segment, from $\mathbf{q}^{i-1}$ to $\mathbf{q}^i$, and then to construct $\mathbf{u}_{pr}$.

This method is more economical computationally and this feature does not conclude the list of its benefits. So, in the previous method a sequence of position vectors is built by the first procedure with no account given to the kinematics. As the result, one may fail at getting $\mathbf{q}$ for a computed $\mathbf{x}$ since no solution is available to $\mathbf{x} = \mathbf{x}(\mathbf{q})$ relating these vectors. In generating the sequence of $\mathbf{q}$ by the modified version, on the contrary, the kinematics is automatically accounted for in the minimization of the functional. If this minimization is performed subject to the constraints imposed on the permissible variations of the robot generalized coordinates, viz., $\mathbf{q}_{min} \leqslant \mathbf{q} \leqslant \mathbf{q}_{max}$, then it will allow for the structure of the robot arm.

As in the previous case, to avoid the gripper moving interruptionally, we have to modify the first procedure in a fashion similar to that employed previously. It is essentially a change in the interpolation technique. Now the ith vector of generalized coordinates is sought by the (i-2)nd, rather than by the (i-1)st, i.e., $\mathbf{q}^i = \mathbf{q}^{i-2} + \Delta\mathbf{q}^{i-1}$.

Both methods described can be modified in order to reduce the possible deviations of the gripper of the ideal path satisfying the imposed constraints. To this end, realization of each ith step (that is, at $\mathbf{x} = \mathbf{x}^i$ and $\mathbf{q} = \mathbf{q}^i$) of motion must be followed by a translation of the gripper into the position where the reaction forces are relaxed to be close to zero. This is done by changing the vector of generalized coordinates so that the deformation of the gripper be zero, that is, the current position vector $\mathbf{x}$ become exactly the deformed one, $\mathbf{x}_{def}$.

To compute the relaxation mode, we need, firstly, to determine the value of $\mathbf{x}_{def} = \mathbf{x}_{def}^{i}$ by (5.8) invoking the reaction forces $\mathbf{G} = \mathbf{G}_m^i$ measured at $\mathbf{q} = \mathbf{q}^i = \mathbf{q}_m$ and the computed undeformed position of the gripper, corresponding to $\mathbf{q} = \mathbf{q}^i = \mathbf{q}_m$, and to have at hand $\mathbf{x}_m = \mathbf{x}(\mathbf{q}_m)$. Secondly, to find the vector corresponding to $\mathbf{x} = \mathbf{x}_{def}^{i}$; this can be achieved by solving the system of kinematic equations at $\mathbf{x}_{def}^{i} = \mathbf{x}(\mathbf{q})$. Thirdly, to compute programmed path $\mathbf{q}_{pr} = \mathbf{q}_{pr}(t)$ and controls $\mathbf{u}_{pr} = \mathbf{u}_{pr}(t)$ at the segment corresponding to $\mathbf{x}$ varying from $\mathbf{x}^i$ to $\mathbf{x}_{def}^{i}$.

5.6 EXTRAPOLATING PROGRAMMED PATH

The next method of programmed path synthesis and control accounts for the constraints in moving a body, so it extrapolates the gripper motion path on the results of its computed position in the previous segments. Accordingly, the multistep algorithm of programmed path and control synthesis has as its first procedure the generation of a position vector $\mathbf{x}^i$ such that (i) is closer to the target than $\mathbf{x}^{i-1}$, and (ii) lies on the curve gained by the path extrapolation and satisfying the constraints. A linear extrapolation brings $\mathbf{x}^i$ onto a direct line of the Euclidean space, which travels through the two points corresponding to the two closest previous deformed positions of the gripper, $\mathbf{x}_{def}^{i-1}$ and $\mathbf{x}_{def}^{i-2}$, which, obviously, belong to the path complaining with the constraints:

$$\mathbf{x}^i = \mathbf{x}_{def}^{i-1} + \mu(\mathbf{x}_{def}^{i-1} - \mathbf{x}_{def}^{i-2}), \tag{5.18}$$

where μ defines the limits of the extrapolation, and $\mathbf{x}_{def}^{i-1}$ and $\mathbf{x}_{def}^{i-2}$ are obtained by (5.8) in terms of measured quantities, $\mathbf{G}_m^{i-1}$ and $\mathbf{G}_m^{i-2}$, and calculated quantities $\mathbf{x}_m^{i-1} = \mathbf{x}(\mathbf{q}_m^{i-1})$ and $\mathbf{x}_m^{i-2} = \mathbf{x}(\mathbf{q}_m^{i-2})$.

Consequently, the first procedure casts into determining by (5.18) the current value of $\mathbf{x} = \mathbf{x}^i$, the distance from $\mathbf{x}^i$ to the set of possible target positions of the gripper $\mathbf{X}_t$, and checking it with the distance for $\mathbf{x}_{def}^{i-1}$. If the distance for $\mathbf{x}^{i-1}$ turns out to be the largest, then the algorithm passes over to the following procedures, namely, computing the vector of generalized coordinates, $\mathbf{q}^i$, by the found $\mathbf{x}^i$, interpolating, and deriving a programmed control. These procedures are identical with that in the first method. Otherwise, $\mathbf{x}$ is sought by the first procedure in the multistep algorithm of the first method.

Instead of extrapolating over the gripper positions domain, one may do this over the domain of the generalized coordinate vectors.

The ordinary, ith value of the generalized coordinate vector is then deduced by

$$\mathbf{q}^i = \mathbf{q}_{\text{def}}^{i-1} + \mu (\mathbf{q}_{\text{def}}^{i-1} - \mathbf{q}_{\text{def}}^{i-2}),$$

where $\mathbf{q}_{\text{def}}^{i-1}$ and $\mathbf{q}_{\text{def}}^{i-2}$ are to be found from (5.13a) by measured quantities $\mathbf{G}_{\text{m}}^{i-1}$, $\mathbf{G}_{\text{m}}^{i-2}$, $\mathbf{q}_{\text{m}}^{i-1}$, and $\mathbf{q}_{\text{m}}^{i-2}$.

This approach cuts the number of procedures required to realize the control by one, however, in turn, it necessitates to narrow the limits of extrapolation defined by μ.

5.7 METHOD OF CONTROL CUTTING AMOUNT OF COMPUTATION

To realize the control by way of the methods considered in the previous sections one would have to resort to a large capacity computer. It would be practically advantageous, therefore, to devise a method to cut the amount of computation even at the cost of a coarser control.

On a way to this objective, we form a control in the form

$$\mathbf{u} = \mathbf{K} (\mathbf{q}_{\text{t}} - \mathbf{q}), \tag{5.19}$$

where $\mathbf{q}_{\text{t}}$ and $\mathbf{q}$ are the target and current (measured) vectors of generalized coordinates, respectively.

The vector $\mathbf{q}_{\text{t}}$ should be formed so that the varying $\mathbf{q}$ comply with the constraints, i.e., $\mathbf{q} = \mathbf{q}_{\text{def}}$. We shall assume that the closed-loop control system described by (5.2), where $\mathbf{K} (\mathbf{q}_{\text{t}} - \mathbf{q})$ is substituted for $\mathbf{u}$, is stable. Seeking for $\mathbf{q}_{\text{t}}$ as in the previous method we can employ extrapolation as follows:

$$\mathbf{q}_{\text{t}}^i = \mathbf{q}_{\text{t},0}^{i-1} + \mu (\mathbf{q}_{\text{t},0}^{i-1} - \mathbf{q}_{\text{t},0}^{i-2}),$$

where $\mathbf{q}_{\text{t},0}^{i-1}$ and $\mathbf{q}_{\text{t},0}^{i-2}$ are the values of $\mathbf{q}_{\text{t}}$ computed on the previous $(i-1)$st and $(i-2)$nd steps, which varies according to the law guiding the body along the path complying with the constraints, $\mathbf{q} = \mathbf{q}_{\text{def}}$.

Let us find an expression to derive $\mathbf{q}_{\text{t},0}$. Substituting for $\mathbf{u}$ in (5.2) its expression from (5.19) and recalling (5.13a) for $\mathbf{G}$ yields for $\mathbf{q}_{\text{t}} = \mathbf{q}_{\text{t},0}$

$$[\mathbf{A} (\mathbf{q}) + \bar{\mathbf{J}}_{\text{a}}] \ddot{\mathbf{q}}_{\text{def}} + \sum_{s=1}^{m} [\dot{\mathbf{q}}^{\text{T}} \mathbf{D}_s (\mathbf{q}_{\text{def}}) \dot{\mathbf{q}}_{\text{def}}] \mathbf{e}_s + [\mathbf{K}_{\text{vf}} + \mathbf{K}_{\text{M}} \mathbf{R}^{-1}] \dot{\mathbf{q}}_{\text{def}}$$

$$+ \mathbf{M}_{\text{fr}} \operatorname{sgn} (\dot{\mathbf{q}}_{\text{def}}) - \mathbf{Q}_{\text{w}} (\mathbf{q}_{\text{def}}) - \mathbf{R} (\mathbf{q}_{\text{def}}) \mathbf{G} - \mathbf{Q}_{\text{d}} = \mathbf{K} (\mathbf{q}_{\text{t},0} - \mathbf{q}_{\text{def}}). \tag{5.20}$$

Subtracting (5.2) at $\mathbf{u} = \mathbf{K}(\mathbf{q}_t - \mathbf{q})$ from (5.20), we get the correction, $\Delta\mathbf{q}_t = \mathbf{q}_{t,0} - \mathbf{q}_t$, for the target vector of generalized coordinates

$$\Delta\mathbf{q}_t = [\mathbf{K}^{-1}\mathbf{R}\mathbf{K}_M^{-1} + (\mathbf{R}(\mathbf{q})\mathbf{C}\mathbf{R}^T(\mathbf{q}))^{-1}]\,\mathbf{R}(\mathbf{q})\,\mathbf{G} + \mathbf{K}^{-1}\mathbf{R}\mathbf{K}_M^{-1}\Delta\mathbf{F} \tag{5.21}$$

where $\Delta\mathbf{F}$ is the difference of the left-hand side of (5.20) and (5.2) less the term $\mathbf{R}(\mathbf{q})\mathbf{G}$.

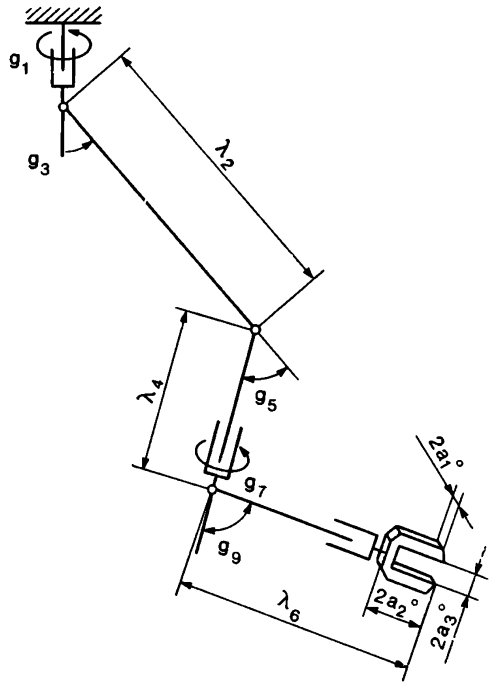


Fig. 5.3 Kinematics of the anthropomorphic robot linkage

Since the flexibility of the gripper joint is normally appreciably high, the term $(\mathbf{R}(\mathbf{q})\mathbf{C}\mathbf{R}^T(\mathbf{q}))^{-1}$ is an order of magnitude or more higher than $\mathbf{K}^{-1}\mathbf{R}\mathbf{K}_M^{-1}$ so that the latter may be neglected along with the term $\mathbf{K}^{-1}\mathbf{R}\mathbf{K}_M^{-1}\Delta\mathbf{F}$ which is small as $\mathbf{q}_{def}$ and $\mathbf{q}$ are usually close. The simplification results in

$$\mathbf{q}_{t,0} = \mathbf{q}_t + \Delta\mathbf{q}_t \approx \mathbf{q}_t + \mathbf{K}^{-1}\mathbf{R}\mathbf{K}_M^{-1}\mathbf{R}(\mathbf{q})\,\mathbf{G} \tag{5.22}$$

and, subject to (5.20), in

$$\mathbf{q}_t^i = (1+\mu)[\mathbf{q}_t^{i-1} + \mathbf{K}^{-1}\mathbf{R}\mathbf{K}_M^{-1}\mathbf{R}(\mathbf{q}^{i-1})\,\mathbf{G}_m^{i-1} - \mu\,[\mathbf{q}_t^{i-2} + \mathbf{K}^{-1}\mathbf{R}\mathbf{K}_M^{-1}\mathbf{R}(\mathbf{q}_m^{i-2})\mathbf{G}_m^{i-2}] \tag{5.23}$$

where $\mathbf{G}_m^{i-1}$, $\mathbf{G}_m^{i-2}$, $\mathbf{q}_m^{i-1}$, and $\mathbf{q}_m^{i-2}$ are the vectors of reaction forces and generalized coordinates measured on the $(i-1)$st and $(i-2)$nd steps of the algorithm.

Obviously, with this method, the computations required to realize the control reduce to deriving a reference signal by (5.23) and a control by (5.19). The amount of these computations is drastically reduced as compared with that of the previously discussed methods. Hence, more simple computation facilities may be implemented.

The vector $\hat{\mathbf{R}}(\mathbf{q})\,\mathbf{G}$ is a component of the vector of generalized forces due to reactions; therefore, it may not be calculated but, instead, be measured by a sensor of forces and torques installed in an arm joint. With this measurement technique, one would have to eliminate, however, from the result those components of generalized force vector due to the weights of links and inertia forces. This can be done whenever pace of arm advance is slow.

To verify experimentally, the discussed methods of control were simulated with the multipurpose dynamic model of robot linkage [21]. The input data corresponded to the linkage of an anthropomorphic robot (Fig. 5.3) whose electromechanical actuators were built around d.c. motors [27]. For the methods discussed in sections 5.4—5.6, the stabilizing component was taken in the form (5.14) with diagonal matrices $\mathbf{K}_1$ and $\mathbf{K}_2$. The input was the programmed values $\mathbf{q}_{pr}(t)$ generated by the methods for a bar moved in supports.

All the verified methods proved feasible. The best results, as regards both the accuracy of guiding along the path and the computational economy, presented the method implementing extrapolation of programmed path. The method of Sec. 5.7 also made a good showing.

List of Symbols

$\mathbf{q}$	vector of generalized coordinates
$\mathbf{x}$	position vector
$\mathbf{u}$	vector of controls
$\mathbf{Q}$	vector of reactions forces
$\mathbf{J}$	matrix of inertia

6 Robot-oriented Software for Computer-assisted Teaching and Control

S. L. Zenkevich, Cand. Sc. (Tech.), and
A. V. Nazarova

6.1 GENERAL

A digital computer controlling the robot functioning allows the operator to conveniently execute control by using it as an interface between input control devices and the manipulator. In control the operator carries on a dialog with an actuator via a computer which generates preprogrammed actions and supports active interface between the operator and robot, which is designed so as to adequately fulfil the directives assigned by the operator. In the following we will discuss a robot-oriented operating system which accomplishes the following functions:

1. Enters and edits motion programs in a dialog mode.
2. Teaches a robot.
3. Executes (or simulates) motion programs formed in advance.
4. Controls a robot by a joystick.
5. Tests the computer-robot communication channels.

The experience of using the operating system (OS) under various conditions has demonstrated its high efficiency which manifests itself in a quick robot readjustment in going from one job to another performed in a dialog mode convenient for the user. The software of motion control and robot teaching ensures wide experimenting in the field of robotic systems.

6.2 DESIGN OF ROBOTIC COMPLEX AND ITS OPERATING SYSTEM

The robotic complex is visualized in Fig. 6.1. The basis of the complex is a control minicomputer having the following characteristics: main storage is 256 Kbytes, word length is 16 bits, swapping time is 1.2 microseconds; variable length of commands.

The actuator used is an electromechanical anthropomorphic six-degrees-of-freedom manipulator. A TV camera integrated into the complex allows a scene image to be entered to the computer memory, the image having a discreteness of 128 × 128 points each of which is characterized by 8 degrees of brightness. All nonstandard external devices are connected by means of CAMAC equipment which provides not only quick swapping with peripherals but also allows the robotic complex to use a wide range of sensors.

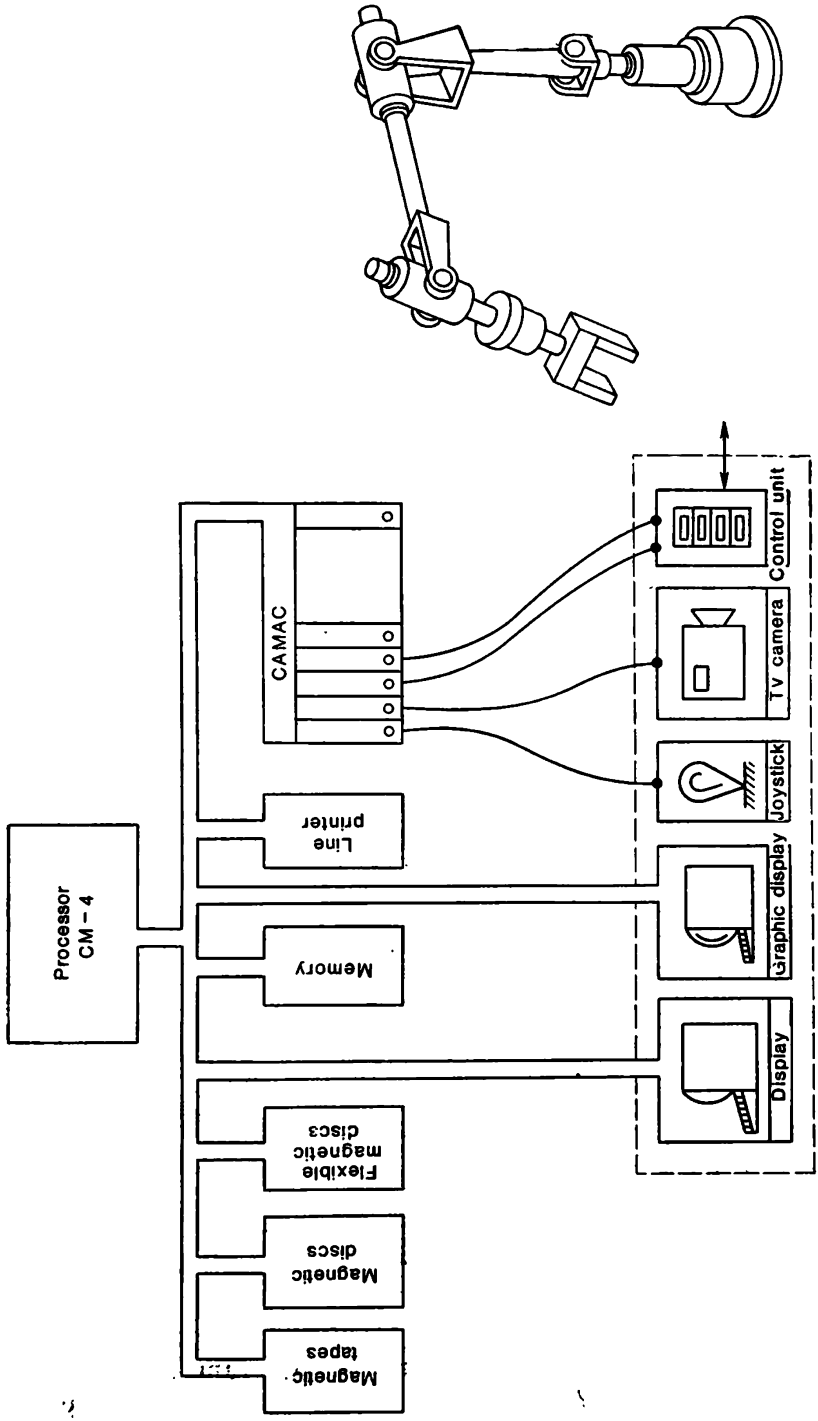


Fig. 6.1 Computer-assisted robotic complex

Before describing the software for motion control and robot teaching, we shall discuss a structure of the operating system and possibilities which it provides for the user. While designing the operating system we proceed from the similarity between the universal and robot-oriented operating systems. This concerns in particular the

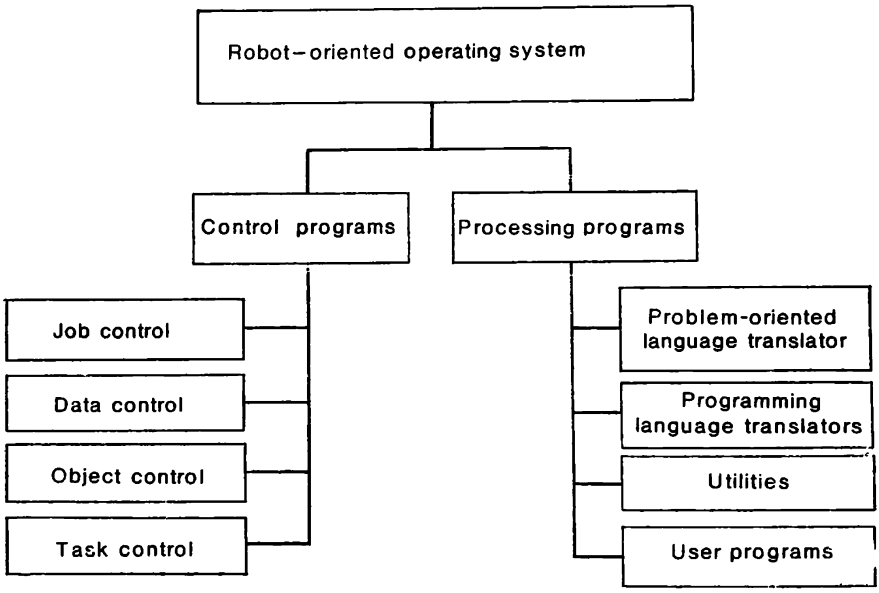


Fig. 6.2 Robot-oriented operating system

control and processing programs comprising job and data control programs and language translators. Fig. 6.2 shows the robot-oriented operating system which in general coincides with that of the universal OS. The operator dialogs with the robot through an alphanumeric display: an interactive monitor receives statements, decodes them and activates appropriate systems programs adequately responding to the instruction received. When necessary the monitor interrogates the operator for additional information, in other words, a system of "prompts" is introduced.

Now we shall briefly discuss the main instructions of the monitor:

```
RUN
FILE NAME=
OPTIONS=
```

This instruction executes the motion program formed earlier. When a program library is available, the operator enters the program

name, and the program executes a complicated succession of robot actions. The statement RUN provides the user with extra possibilities to simulate the actuator motion using to this end the graphic display screen.

```
JOYSTICK
```

This instruction uses the joystick as an input control device. In this mode no permanent interrogation or entry of signals comes from the joystick. The instruction itself initiates various programs by processing this information and prepares an interruption vector. Such a technique of control by the joystick overrides the other modes of control (the most appropriate in this respect is a teaching mode, see below), since the joystick, when engaged, entails interruptions.

```
PROGRAM
INPUT SOS-FILE=
OUTPUT SOS-FILE=
OUTPUT RUN-FILE=
OPTIONS=
```

The instruction controls the robot by means of the stored program. This comprises a text written in a source language of the motion control. The text can be entered by the operator from the display keyboard or formed earlier and stored in the external memory.

The instruction PROGRAM enables the user to debug (by-step) a task program or to simulate the sequence of robot positions by displaying them on the screen. In greater detail the language of motion control will be described below.

```
EDIT
INPUT SOS-FILE=
OUTPUT SOS-FILE=
```

The edit instruction edits the existing text file or forms a new file in the external memory.

```
SMON
MODE=
```

This multifunctional instruction solves a number of problems on control and teaching of the robot in a dialog mode:

— forms a robot motion program comprising a sequence of arm positions. (This can be done in a dialog mode from a display keyboard or in a teaching mode; moreover, this program can be formed on an external medium);

— automatically executes the program formed;
— provides the operator with the information on the current position of the actuator and also on the motion program being run;
— prepares a motion program of the minicomputer-controlled manipulator.

Before considering in detail some modes provided by the SMON instruction we would like to comment on the following. Whatever method is used to introduce a positioning point, it is always characterized by the same set of parameters, such as: point number, six-dimensional vector of coordinates (it can be represented by two

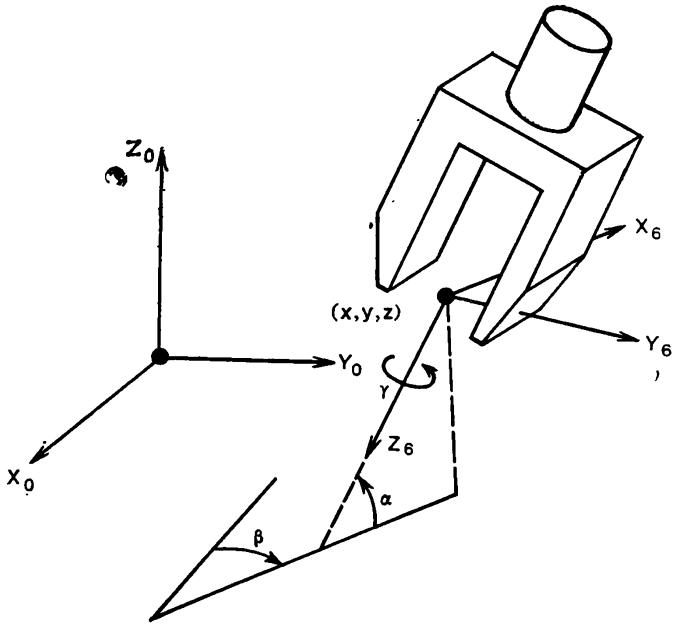


Fig. 6.3 Hand frame

methods: either by means of angles in joints q_1, q_2, ., q_6 or through coordinates $x, y, z, \alpha, \beta, \gamma$, Fig. 6.3); and an additional set of parameters (V is the linear speed of approaching the point; OM is the angular speed of approaching the point; T is time the hand is staying at the point; JAW is the parameter characterizing the hand status).

Now we shall proceed with describing some modes the SMON instruction provides for the user:

1. TT

This mode forms a motion program as an enumerated succession of positioning points using the terminal as an input device. The program is formed in a dialog by answering the questions displayed on the screen.

An example of such a dialog in forming a simple program is shown in Fig. 6.4.

```
MNT>SMON
MODE=TT
NUMBERS OF CICLES [I2]=1
"S" OR "Q" INPUT [S/Q]=Q
Q  1=30. Q2=-30. Q3=45. Q4=90. Q5=-45.5 Q6=0.
JW=0 T=3. V=0.15 OM=0.3
CONTINUE TT [Y/N]=N
MODE=
```

Fig. 6.4 Forming a program

2. DK

In this mode the motion program is output/input on an external medium; in this way we can form a library containing programs of standard motions.

3. TC

The mode TC teaches the robot using the display keyboard as a pushbutton control device. By entering letter instructions the user actuates the robot. When the state achieved by the arm agrees with the operator's wish, he can memorize it thus including the positioning point to the motion program being formed.

All the instructions entered in the TC mode can be divided as follows.

(a) Hand rotation about the axes of the hand-tied frame (the linear coordinates of the characteristic point of the hand (see Fig. 6.3) remain constant):

A = positive rotation about the X_6 axis;

B = negative rotation about the X_6 axis;

F = negative rotation about the Z_6 axis.

(b) Hand sliding along the axes of the inertial and tied frames (the hand orientation remains constant):

G = positive sliding along the Z_0 axis ("upward");

H = negative sliding along the Z_0 axis ("downward");

K = positive sliding along the X_6 axis;

P = negative sliding along the Z_6 axis.

(c) Service instructions:

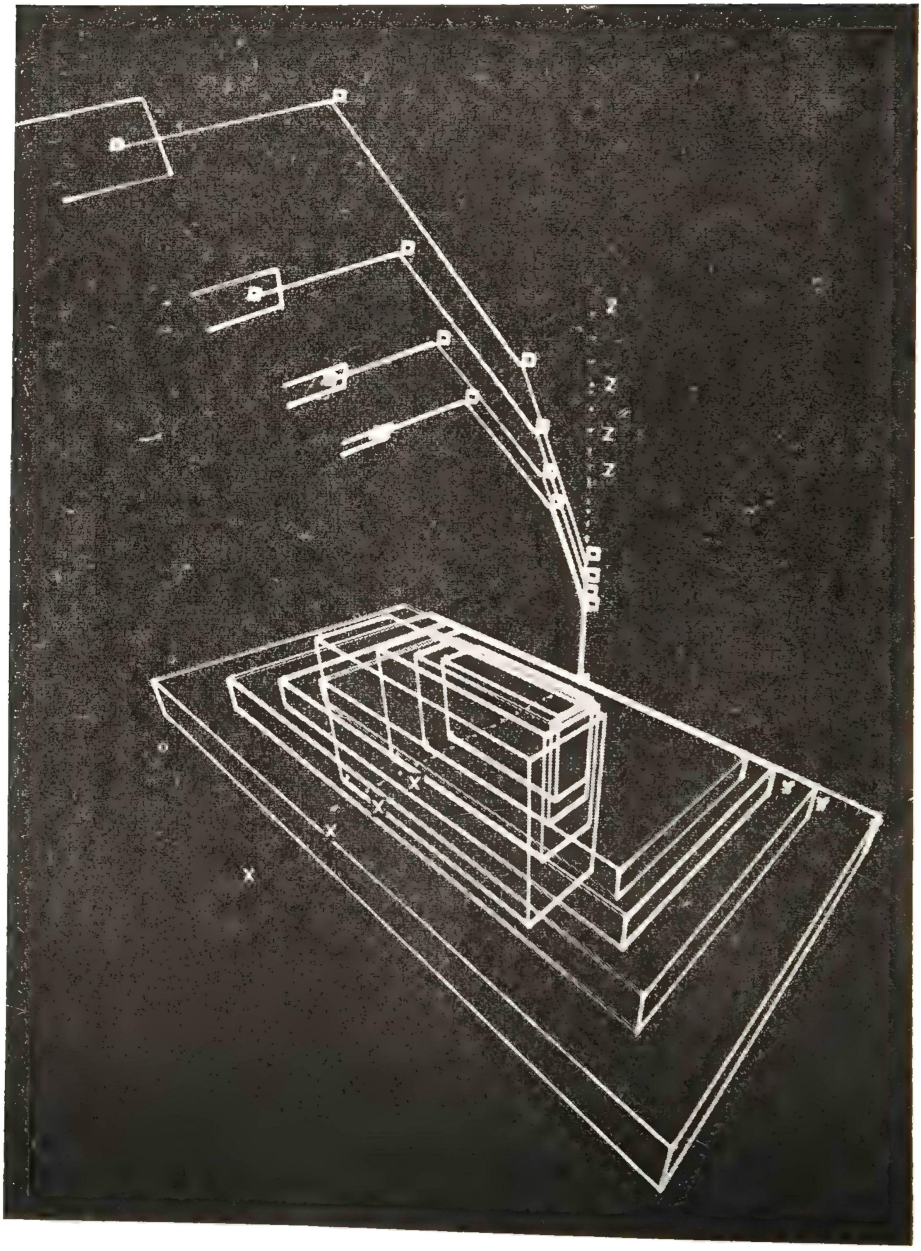


Fig. 6.5a Observer's position: "Nearer"

Q = remember the current state of the arm and include it into the task program;

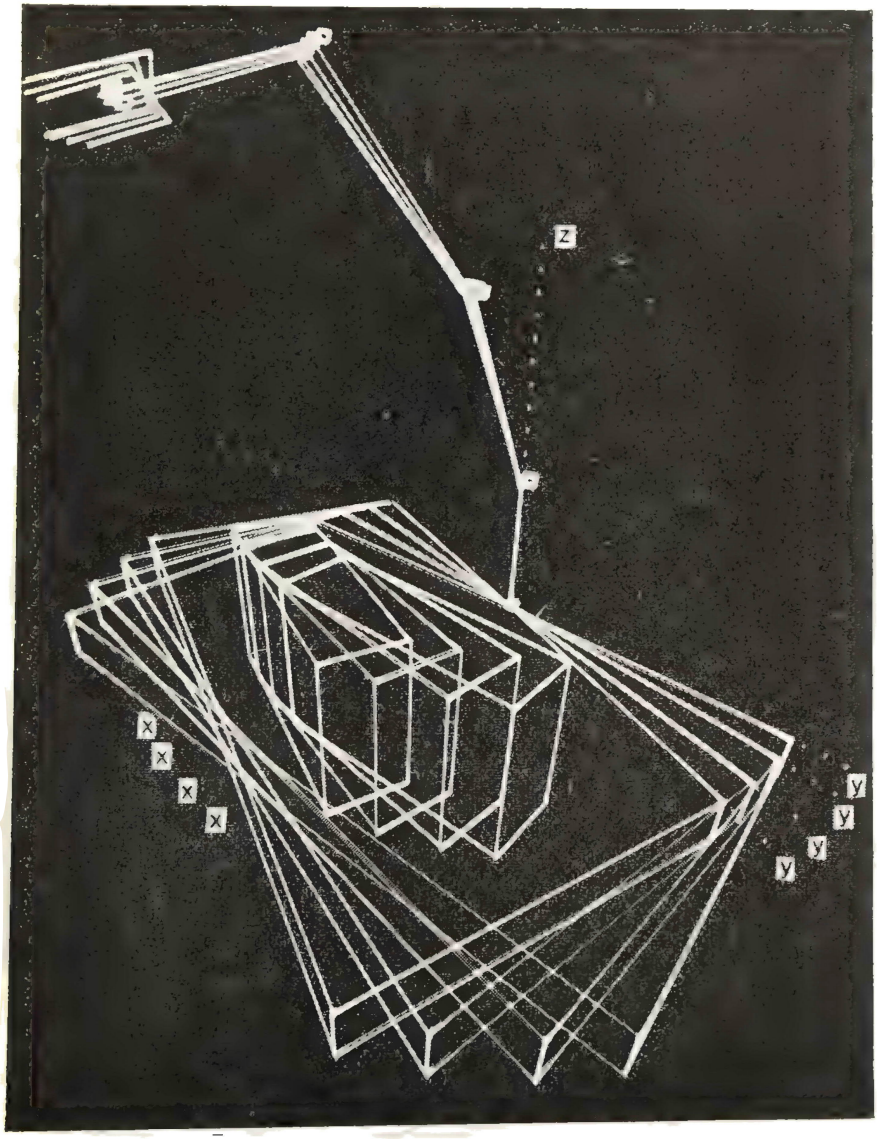


Fig. 6.5b "To the Left"

R = open the hand;
S = close the hand;
X = go out of the teaching mode.

The set of instructions given above sufficiently quickly and effectively moves the robot hand to a required position.

4. EX

The EX mode automatically executes the program formed, the motion may be simulated on the graphic display screen.

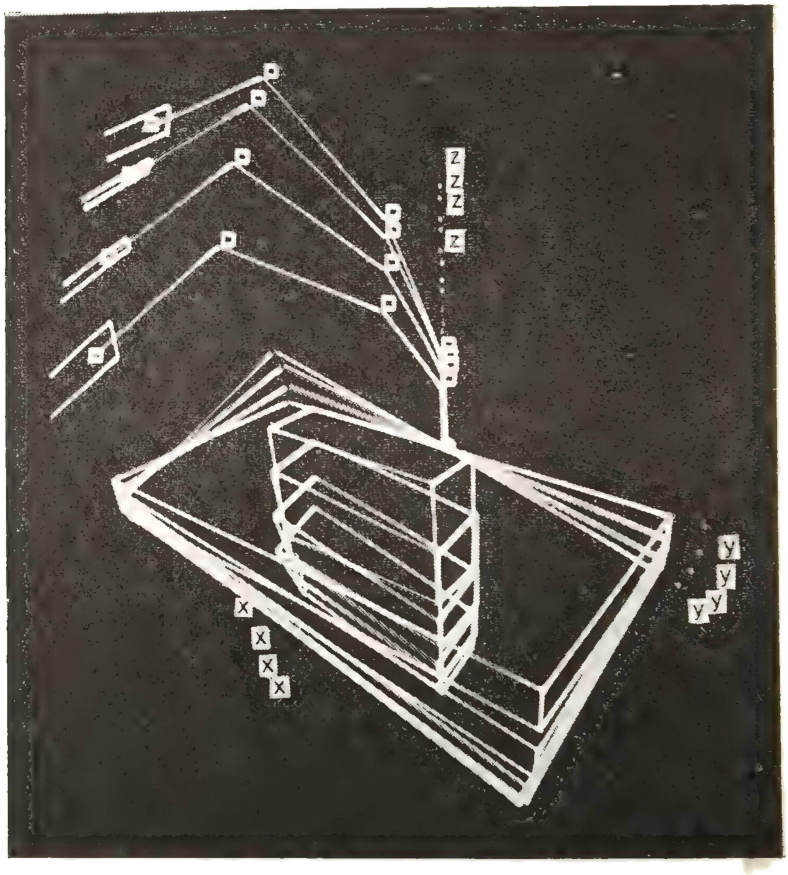


Fig. 6.5c "Higher"

5. TQ

In this mode the user controls each joint separately.

In addition to the modes listed above the SMON instruction carries out the service operations that read out the current status of the actuator and check the working program being formed.

TEST

This instruction of the monitor tests the communication channels between the computer and the actuator. Testing is identifying para-

meters of analog/digital and digital/analog converters and some parameters of the joint drives. This occurs in the experiment which

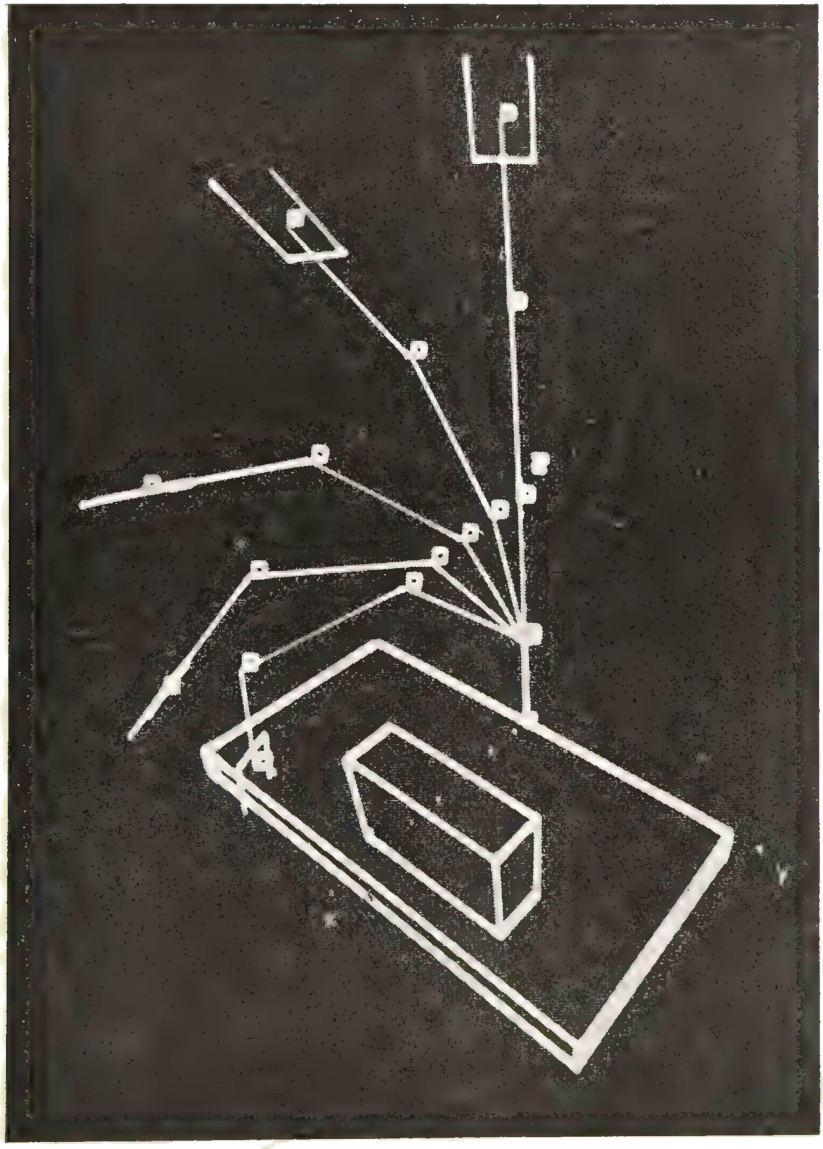


Fig. 6.6 Sequence of arm positions in gripping a parallelipiped

involves directing a sequence of control voltages to the drives, measuring the rotation angles of the corresponding joints and processing the results obtained.

6. TV

This instruction displays on a screen a world fragment in which the robot and its actuator function. Also displayed on the screen is a menu which allows the operator to transform the scene by a light pen, in particular to select an adequate place for observation and to indicate the position which the robot hand should take. Its corresponding positions are shown in Figs. 6.5-6.7. Fig. 6.6 shows a sequence of arm positions in gripping a parallelepiped.

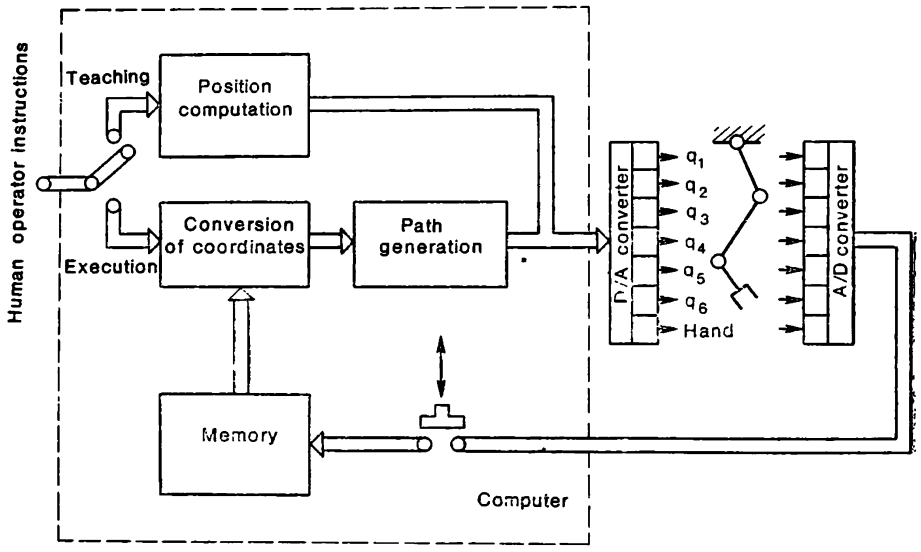


Fig. 6.7 Diagram of controlling the robot motion

The TV statement simulates the actuator behavior in the mode of assigning a succession of positioning points from a display console or by means of a light pen.

Thus we have described some monitor instructions which enable the user to control and teach the robot. Mathematical models and algorithms taken as a basis for the software of robot teaching and control will be described below.

6.3 ROBOT MOTION CONTROL LANGUAGE

To move a robot in an incompletely predicted situation requires a motion program that takes into account not only the sequence of positional points but also the conditions of motion from one fixed point to another. To solve the problem the designer needs a source problem-oriented language, the basic elements of which include both statements controlling the motion and statements processing sensor information and controlling the program run.

At present motion control languages are booming just as were programming languages some years ago. Nowadays a diversity of programming languages (ALGOL, FORTRAN-IV, PL/1) cover the greater portion of the user's needs. It is rather optimistic, however, to anticipate the same with the motion control languages.

Now we shall briefly describe the main elements of the source problem-oriented language of the robot motion control.

Line Format. This control program is a sequence of statements each of which may be identified by label including not more than five decimal numbers. The line format mainly coincides with the format used in FORTRAN-IV but for that in the first position of the line letter F is permissible. It means that the line is a statement written in the FORTRAN-IV language which enables the user to form some specific rules of controlling the motion or processing data.

Source Language Statements. These include executable and non-executable ones. The executable statements result either in some displacement by an actuator or in preparation for this displacement by initiating sensors, reading-out and processing corresponding information. The executable statements also transfer the control. The nonexecutable statements inform the translator of any peculiarities of the program being entered and do not cause any action of the robot or sensors.

Any motion of the actuator can be represented by a sequence of some elementary motions. Therefore it is desirable to have not a range of programs for each possible motion of the actuator but rather a set of statements which being differently grouped can ensure any motion. It would be extremely inconvenient, however, to use this limited set of elementary statements for assigning a motion program. That is why while entering a program the user employs not the elementary statements but those reduced to such. They are called macrostatements.

Before we pass on to discussing some elementary statements it should be noted that the actuator position is assigned by one of the following methods:

(i) by a vector the components of which are either angles in joints

$\mathbf{q} = [q_1, q_2, \ldots, q_6]^T$

or hand Cartesian coordinates

$\mathbf{S} = [x, y, z, \alpha, \beta, \gamma]^T,$

(ii) by a **T** matrix of the following structure

$$\mathbf{T} = \begin{bmatrix} \mathbf{e}_{x6} & \mathbf{e}_{y6} & \mathbf{e}_{z6} & \mathbf{p}_6 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

where $\mathbf{e}_{x6}$, $\mathbf{e}_{y6}$, $\mathbf{e}_{z6}$ are directional cosine vectors in the inertial coordinate system $(0_0X_0Y_0Z_0)$ with reference to the hand-tied coordinate

system ($0_6X_6Y_6Z_6$) (see Fig. 6.3); $\mathbf{p}_6$ is the vector connecting the origin of the inertial frame with that of the hand-tied frame.

Now we shall enumerate some of the elementary statements of the source language and describe their operation using the following system of notations: the numbers are written in small letters, the names (identifiers) in capital letters.

The statements are represented in the form:

1. PQ (j, q_1, q_2, q_3, q_4, q_5, q_6).

The statement gives control signals to the actuator

$$q_i^g = q_i + jq_i^a, \quad i = 1, 2, \quad 6,$$

where q_i^g are control signals; q_i^a are the ith joint angles, and

$$j = \begin{cases} 1, \text{ control angles are relatively assigned} \\ 0, \text{ control angles are absolutely assigned.} \end{cases}$$

2. IPQ (j, A)

This statement is equivalent to the preceeding one differing in that the control angles are assigned not directly as in the previous case but as belonging to the array A, that is

$$A\ (i) = q_i, \quad i = 1, 2, \quad ., 6,$$

where A is an identifier determining a 6-dimensional array.

3. PQM (j, m, q)

This statement directs a signal to the mth drive

$$q_m^g = q + jq_m^a,$$

where j and q_m^a are the same as for the PQ statement.

We shall note that to reach a terminal point executed should be the statement PQ (j, q_1, q_6) or the following sequence of the PQM statements

PQM (j, 1, q_1),
PQM (j, 2, q_2),

.

PQM (j, 6, q_6).

Obviously the pathways to the target will be different.

4. IPQM (j, m, A)

This statement is the same as PQM, though A is not a number but the name of a variable containing a desired angle (A is a scalar).

Now we shall describe some statements which execute arithmetic operations in solving kinematic problems, and some additional operations.

5. QST (Q, T)

This statement solves the primal problem, i.e. it finds the hand Cartesian coordinates by the assigned angles in the joints. Here Q is a 6-dimensional array comprising these angles

$$Q(i) = q_i,\ i = 1, 2, \ldots, 6,$$

and T is a 4×4 matrix completely characterizing the hand state

$$\mathbf{T} = \begin{vmatrix} \mathbf{e}_{x6} & \mathbf{e}_{y6} & \mathbf{e}_{z6} & \mathbf{p}_6 \\ 0 & 0 & 0 & 1 \end{vmatrix}.$$

6. SQ (n, S, Q)

The statement solves a nonlinear inverse problem, i.e. it determines the angles in the joints q_i belonging to the array Q by the assigned hand state S

$$\mathbf{S} = [x, y, z, \alpha, \beta, \gamma]^T.$$

The parameter n characterizes a branch while solving the inverse problem.

7. YSQ (DS, Q, DQ, QA)

This statement solves a linearized inverse problem. Its array DS comprises an increment in the 6-dimensional vector of the arm state; Q are the desired angles in the joints; DQ are the increments in these angles and QA are the angles in the joints. Thus

$$Q = QA + DQ.$$

8. TR (T, S)

This statement transforms matrix T into a vector

$$\mathbf{S} = [x, y, z, \alpha, \beta, \gamma]^T.$$

9. RT (S, T)

This statement transforms vector S into matrix T.

10. ASK (Q)

The statement transfers the angles in manipulator joints to the array Q.

The following statements control the program running.

11. JMP (n)

The statement unconditionally transfers control to the label n.

12. IT (m, n)

It transfers control to the label n, if during further run of the working program the hand touches some object, which causes an activation of tactile sensors masked by the number m.

13. IT (m, 0)

It cancels the statement IT (m, n).

Macrostatements. A set of elementary statements can execute any motion of an actuator. For convenience, however, the user should

unite these elementary statements into standard macrostatements which control complicated motions of the robot hand.

A macroprocessor coming across a macrostatement in the source text replaces it (widens it up) with some text, taken from the library, in a source control language which is then directed to the translator input [25]. We illustrate by the following statements. (The names generated by the macroprocessor itself are preceded by a symbol "@".)

1. OP, CL

It results in opening (closing) the hand. Its macroexpansion can be presented in the form

OP→ PQM (0, 7, 0) CL→ PQM (0, 7, 1)
WAIT (3.) WAIT (3.)

2. PNT (T)

The statement transfers the hand to the position characterized by matrix **T**

$$\mathbf{T} = \begin{vmatrix} \mathbf{e}_{x6} & \mathbf{e}_{y6} & \mathbf{e}_{z6} & \mathbf{p}_6 \\ 0 & 0 & 0 & 1 \end{vmatrix}.$$

The macrostatement PNT (T) has the following macroexpansion:

PNT (T)→ TR (T, @ S) conversion of matrix **T** into vector @ **S**;
SQ(0, @S, @Q) solution of a nonlinear inverse problem. Here vector @ **Q** comprises the desired angles of rotation in the joints.
IPQ (0, @ Q) generation of control signals to the drives.

3. PNS (S)

The statement transfers the hand into a position characterized by the array **S**

$$\mathbf{S} = [x,\ y,\ z,\ \alpha,\ \beta,\ \gamma]^T$$

Its macroexpansion has the following form:

PNS (S)→ SQ (0, S, @Q) solution of nonlinear inverse problem;
IPQ (0, @Q) generation of control signals to the drives.

Language translator is a text macroprocessor which converts statements written in the motion control source language to the format of the FORTRAN-IV language.

Here are the examples of some statements discussed above:

1. PQ (j, q_1, q_2, . . ., q_6)

The statement is converted to the format of FORTRAN-IV language, i.e.

ARXY (1) = q_1
ARXY (2) = q_2
ARXY (3) = q_3
ARXY (4) = q_4

ARXY (5) = q_5
ARXY (6) = q_6
CALL IPQ (j, ARXY).

The first statement PQ generates a line in the program DIMENSION ARXY (6).

2. IPQ (j, A)

The statement is changed into a text
CALL IPQ (j, A).

3. EOT

The statement informing of the end of the entered text is expanded in the following way:
CALL CHAIN (FLNM, 0)
END

The subprogram CHAIN loads the monitor and shifts control thereto.

6.4 MOTION CONTROL SOFTWARE

Motion control is visualized in Fig. 6.7. It accomplishes the following functions:

—synthesizes hand paths;

—finds corresponding angles in joints (solves the inverse problems);

—generates control signals to the actuators using a spline interpolation.

The system programs form a task program comprising a set of data completely describing positioning points. As the initial step the program plans the motion from one positioning point to another, which insures an assigned hand path. Here the path is a straight line connecting two successive positions.

Let $\mathbf{T}_s$ and $\mathbf{T}_f$ be the matrices determining generalized coordinates in two successive positions

$$\mathbf{T}_s = \left[\begin{array}{c|c} \mathbf{l}_s & \mathbf{p}_s \\ \hline 0\ 0\ 0 & 1 \end{array}\right] \text{ and } \mathbf{T}_f = \left[\begin{array}{c|c} \mathbf{l}_f & \mathbf{p}_f \\ \hline 0\ 0\ 0 & 1 \end{array}\right], \tag{6.1}$$

where $\mathbf{l}_s$, $\mathbf{l}_f$ are the matrices of direction cosines of a relative rotation corresponding to the initial and final positions; $\mathbf{p}_s$, $\mathbf{p}_f$ are the transfer vectors, i.e. the vectors specified in the Cartesian inertial system.

The purpose of the planning is to divide the line segment connecting two adjacent points and to find a $\mathbf{T}_i$ matrix corresponding to each division node. The procedure of solving this problem is as follows [5]:

(i) To find the linear and angular distances between the points (the linear distance p is to be found in a trivial way; in our case we

consider the motion to occur along a straight path), i.e.

$$p = \| \mathbf{p}_f - \mathbf{p}_s \|. \tag{6.2}$$

To find the angle φ and direction (characterized by a unit vector $\mathbf{e}$) about which the rotation occurs we shall introduce a matrix $\mathbf{B} = \mathbf{e}_f \cdot \mathbf{e}_s^T$

$$\cos\varphi = \frac{1}{2}(B_{11}+B_{22}+B_{33}-1), \quad (\varphi \subset [0, \pi]),$$
$$\mathbf{c} = \frac{1}{2\sin\varphi}(B_{32}-B_{23},\ B_{13}-B_{31},\ B_{21}-B_{12})^T. \tag{6.3}$$

(ii) To find the number of division nodes from the assigned maximum linear (Δp) and angular ($\Delta\varphi$) displacements along each division element (we shall note that these displacements determine the precision with which the hand describes the straight path)

$$m = \max([\varphi/\Delta\varphi], [p/\Delta p]) + 2, \tag{6.4}$$

where bracketing denotes an operation of taking an integral part.
(iii) To find, finally, the succession of matrices

$$\mathbf{T}_i = \left|\begin{array}{c|c} \mathbf{l}_i & \mathbf{p}_i \\ \hline 0\ 0\ 0 & 1 \end{array}\right|, \quad i = 1, 2, \ldots, 6$$

determining the location and orientation of the hand in the ith division point. We shall note that $\mathbf{T}_1 = \mathbf{T}_s$, $\mathbf{T}_m = \mathbf{T}_f$. The matrix $\mathbf{l}_i$ and the vector $\mathbf{p}_i$ are as follows:

$$\mathbf{l}_i = B_i \mathbf{l}_1,$$
$$\mathbf{p}_i = \mathbf{p}_1 + (i-1)\frac{\mathbf{p}_m - \mathbf{p}_1}{m-1}, \tag{6.5}$$

where

$$B_i = \cos\varphi_i \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix} + (1-\cos\varphi_i) \begin{vmatrix} c_1^2 & c_1c_2 & c_1c_3 \\ c_2c_1 & c_2^2 & c_2c_3 \\ c_3c_1 & c_3c_2 & c_3^2 \end{vmatrix}$$
$$+ \sin\varphi_i \begin{vmatrix} 0 & -c_3 & c_2 \\ c_3 & 0 & -c_1 \\ -c_2 & c_1 & 0 \end{vmatrix}$$

$$\varphi_i = (i-1)\frac{\varphi}{m-1}.$$

(iv) To find time of hand motion from the initial to the final point

$$t = \max(\varphi/\omega,\ p/v), \tag{6.6}$$

where ω and v are the parameters characterizing the target positioning point.

Now we shall turn to solving a kinematic problem. The planning algorithm gives a succession of matrices $\mathbf{T}_i$, $i = 1, 2, \ldots, m$ insuring straight motion of the hand. The next step is to solve an inverse problem for each of the points obtained. As algorithms of kinematic problems are often oriented at a particular kinematic scheme we shall not go into detail while considering this problem, but only give a final result, i.e. a set of angles in the joints

$$\{\mathbf{q}_i\},\ i = 1, 2, \ldots, m.$$

Here $\mathbf{q}_i = [q_{i1}, q_{i2}, \ldots, q_{i6}]^{\mathrm{T}}$ insures a required arm state at each division node. (For more detail refer to [43].)

To provide a preplanned motion of the hand along a required path assigned by m points in the space of the generalized coordinates a sufficiently smooth signal $q(t)$ is to be applied to each of the actuator drives. This function should be continuous and have continuous derivative with two discontinuities at most. Taken as such functions [47] are cubic splines which represent cubic polynomials for each section

$$P_{i3}(t) = a_{0i} + a_{1i} \cdot t + a_{2i} \cdot t^2 + a_{3i} \cdot t^3,\ t \subset [t_{i-1},\ t_i]. \tag{6.7}$$

The approximating control signals $q_k(t)$ (where k is the number of joints) must be limited both by speed and by acceleration

$$|\dot{q}_k(t)| \leqslant v_k$$

$$k = 1, 2, \ldots, 6$$

$$|\ddot{q}_k(t)| \leqslant a_k$$

over the entire range of control.

The problem consists in finding coefficients determining approximating polynomials over all intervals of division and over all degrees of freedom. These coefficients should meet the condition of limited speed and acceleration and ensure continuity of the first and second derivatives in the division nodes [43].

6.5 TEACHING OF ROBOT

In the teaching mode, the human operator leads the robot through a series of simple displacements along the assigned axes, the hand orientation and rotation about these axes and coordinates of the hand remain unchanged.

Such motions are realized on the following basis. Let $\mathbf{T}$ be a matrix determining the current position of the hand

$$\mathbf{T} = \begin{vmatrix} \mathbf{e}_x, & \mathbf{e}_y, & \mathbf{e}_z, & \mathbf{p} \\ 0 & 0 & 0 & 1 \end{vmatrix}.$$

The rotation angles and displacements being small, the hand elementary motions will make the matrix **T** change and have the following form:

(i) Rotation about the axes of the tied coordinate system through an angle φ
about the X axis

$$\mathbf{T}' = \begin{vmatrix} \mathbf{e}_x & \mathbf{e}_y + \varphi\mathbf{e}_z & \mathbf{e}_z - \varphi\mathbf{e}_y & \mathbf{p} \\ 0 & 0 & 0 & 1 \end{vmatrix},$$

about the Y axis

$$\mathbf{T}' = \begin{vmatrix} \mathbf{e}_x & -\varphi\mathbf{e}_z & \mathbf{e}_y & \mathbf{e}_z + \varphi\mathbf{e}_x & \mathbf{p} \\ 0 & 0 & 0 & 1 \end{vmatrix},$$

about the Z axis

$$\mathbf{T}' = \begin{vmatrix} \mathbf{e}_x + \varphi\mathbf{e}_y & \mathbf{e}_y - \varphi\mathbf{e}_x & \mathbf{e}_z & \mathbf{p} \\ 0 & 0 & 0 & 1 \end{vmatrix}.$$

(ii) Sliding along the axes of the tied coordinate system for a distance r
along the X axis

$$\mathbf{T}' = \begin{vmatrix} \mathbf{e}_x & \mathbf{e}_y & \mathbf{e}_z & \mathbf{p} + r\mathbf{e}_x \\ 0 & 0 & 0 & 1 \end{vmatrix},$$

along the Y axis

$$\mathbf{T}' = \begin{vmatrix} \mathbf{e}_x & \mathbf{e}_y & \mathbf{e}_z & \mathbf{p} + r\mathbf{e}_y \\ 0 & 0 & 0 & 1 \end{vmatrix},$$

along the Z axis

$$\mathbf{T}' = \begin{vmatrix} \mathbf{e}_x & \mathbf{e}_y & \mathbf{e}_z & \mathbf{p} + r\mathbf{e}_z \\ 0 & 0 & 0 & 1 \end{vmatrix}.$$

(iii) Sliding along the Z axis of the fixed coordinate system for a distance r

$$\mathbf{T}' = \begin{vmatrix} \mathbf{e}_x & \mathbf{e}_y & \mathbf{e}_z & \mathbf{p}' \\ 0 & 0 & 0 & 1 \end{vmatrix},$$

where $\mathbf{p}' = (p_x, p_y, p_z + r)^{\mathrm{T}}$.

Then, after the corresponding matrix $\mathbf{T}'$ is formed in response to the instruction entered by the operator, the angles in the joints are to be found (i.e. a kinematic problem is to be solved), and those angles are directed to the drives as control signals.

6.6 TESTING OF COMMUNICATION CHANNELS

In controlling the actuator, the digital/analog converter transforms control signals $q(t)$ into $\bar{q}(t)$ and applies them to the drives. The precision of this transformation finally determines the operation precision of the entire system. One of the test problems is to find the correspondence between the input controls applied to each joint and the angle of rotation of this joint.

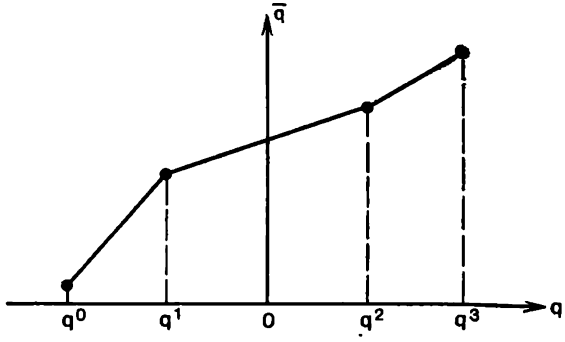


Fig. 6.8 Piecewise-linear approximation of the input/output function

In the following we shall assume piacewise-linear approximation of the input/output function (Fig. 6.8), i.e.

$$\bar{q} = a^i q + b^i;\ q \subset [q^{i-1},\ q^i],\ i = 1,\ 2,\ \ldots,\ n, \tag{6.8}$$

where q is a continuous rotation angle of the joint under consideration; $\bar{q}$ is the same expressed in a digital form.

To determine a set of coefficients $(a^i,\ b^i)$ we use a theory of recurrent estimation [48]. To this end we write the input/output equations in the form

$$q = \alpha^i \bar{q} + \beta^i,\ \bar{q} \subset [\bar{q}^{i-1},\ \bar{q}^i],\ i = 1,\ 2,\ \ldots,\ n, \tag{6.9}$$

where α^i, β^i, a^i and b^i are related as follows

$$\alpha^i = 1/a^i,\ \beta^i = -b^i/a_i.$$

Then the equations of the object and measurements will assume the form

$$\alpha_k^i = \alpha_{k-1}^i, \tag{6.10}$$

$$\beta_k^i = \beta_{k-1}^i,$$

$$Z_k^i = \alpha_k^i \cdot \bar{q}_k + \beta_k^i + \xi_k, \tag{6.11}$$

where ξ_k is a random δ-correlated normally distributed sequence expressing a precision of the movable joint

$$\operatorname{cov}(\xi_k, \xi_l) = \sigma^2 \delta_{kl},$$

$$\delta_{kl} = \begin{cases} 1 & k = l \\ 0, & k \neq l, \end{cases} \tag{6.12}$$

Next, we shall fix the ith segment of the approximated straight line and introduce the following notations:

$\mathbf{x} = (x_1, x_2)^{\mathrm{T}} = (\alpha^i, \beta^i)$ is a vector of phase coordinates;

$\mathbf{H} = (\bar{q}_k, 1)$ is a matrix of measurements;

$\hat{\mathbf{x}} = (\hat{x}_1, \hat{x}_2)$ is a vector of estimating phase coordinates;

$\mathbf{P} = \begin{vmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{vmatrix}$ is a matrix of posterior covariances of the estimate errors.

Then the algorithm of parameter evaluation can be presented in the form

$$\begin{aligned} \hat{\mathbf{x}}_k &= \hat{\mathbf{x}}_{k-1} + 1/\sigma^2 \mathbf{P}_k \mathbf{H}_k^{\mathrm{T}} (z_k - \mathbf{H}_k \hat{\mathbf{x}}_{k-1}); \\ \mathbf{P}_k &= \mathbf{P}_{k-1} - \mathbf{P}_{k-1} \mathbf{H}_k^{\mathrm{T}} (\mathbf{H}_k \mathbf{P}_{k-1} \mathbf{H}_k + \sigma^2)^{-1} \mathbf{H}_k \mathbf{P}_{k-1}. \end{aligned} \tag{6.13}$$

The initial values $\hat{\mathbf{x}}_0$ and $\hat{\mathbf{P}}_0$ in the expression (6.13) are parameters α and β of the corresponding covariance matrix. They are calculated from the line segment connecting the first two points $(\bar{q}_0, q_0)$, $(\bar{q}_1, q_1)$. We turn to the next line segment when the approximation error exceeds an allowable one, i.e. at an nth step the following inequality is satisfied

$$|\hat{\alpha}_{n-1} \bar{q}_n + \hat{\beta}_{n-1} - z_n| > \varepsilon. \tag{6.14}$$

6.7 COMPUTER GRAPHICS

The consideration of mathematical relations allows us to create algorithms of machine graphics ensuring a display of the scene on a screen and its transformation by means of a light pen. The machine graphics is based on the use of homogeneous coordinates and also on the algorithm of central projecting, cutting and removing of invisible lines [33].

Let some point in the robot world fragment be described by a vector specified by homogeneous coordinates $(x, y, z, 1)^{\mathrm{T}}$. To calculate the coordinates of the point projection on the display screen

it is necessary to pass on to the observer's coordinate system (Fig. 6.9) and then to carry on a projection operation (and if neces-

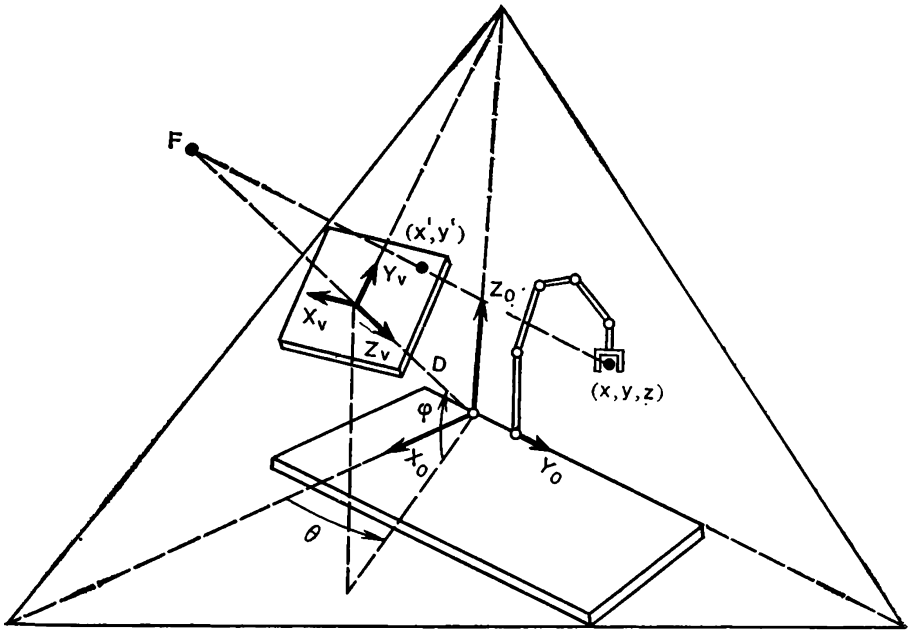


Fig. 6.9 Robot and observer's frames

sary, a scaling operation). This conversion is carried out as follows:

$$\begin{vmatrix} x't \\ y't \\ z't \\ t \end{vmatrix} = \mathbf{MPT}_D\mathbf{T}_x\mathbf{T}_y\mathbf{T}_0 \begin{vmatrix} x \\ y \\ z \\ 1 \end{vmatrix},$$

where

$$\mathbf{T}_0 = \begin{vmatrix} 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = \text{change of coordinate axes,}$$

$$\mathbf{T}_y = \begin{vmatrix} \cos\theta & 0 & -\sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ \sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = \text{rotation about the } Y \text{ axis through an angle } \theta,$$

$$\mathbf{T}_x = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\varphi & \sin\varphi & 0 \\ 0 & -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = \text{rotation about the } X \text{ axis through an angle of } \varphi,$$

$$\mathbf{T}_D = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & D \\ 0 & 0 & 0 & 1 \end{vmatrix} = \text{sliding along the } Z \text{ axis for a distance of } D,$$

$$\mathbf{P} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/f & 1 \end{vmatrix} = \text{central projecting},$$

$$\mathbf{M} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & m \end{vmatrix} = \text{scaling}.$$

To transform the inertial frame to that of the observer's, recourse is had to the matrix $\mathbf{T} = \mathbf{T}_D\mathbf{T}_x\mathbf{T}_y\mathbf{T}_0$, while to project and scale the image having the dimensions of a display screen use is made of the matrices **P** and **M**.

The change in the observer's position is realized (as has been stated above by means of a light pen) by changing parameters θ, φ and D entering the transformation matrices.

6.8 CONCLUSION

The proposed robot-oriented operating system widens the user capabilities of controlling the robot. The availability of a computer in a control circuit allows the operator to effectively form an actuator motion program both in dialog and teaching modes. The testing of the system operation has proved its high serviceability.

List of Symbols

q	vector of generalized coordinates
S	vector of hand state
T	matrix determining the hand state and relating the hand-tied frame to the inertial frame

7 Methods for Evaluating Robots' Software

P. I. Rudakov, Cand. Sc. (Tech.)

7.1 GENERAL

Beyond the obvious objectives of minimum computation time and memory required, a robot control software should be devised so as to cover a great range of programs, calling for minimum readaptation in going from one task to another. The pertinent procedure of readaptation should be simple as should be that of program debugging and monitoring.

Depending upon the problem at hand, be it a robot control, communication facilities development, decision making in a complex situation, or the program generation for a computer-augmented manipulator, the designer has to choose among the above objectives a single one or a few most advantageous. If a single objective cannot be stick at, then the development of an efficient software becomes a more hard task, since to test a program for performance, the designer should devise a generalized, normally complex, performance criterion.

7.2 PATTERN RECOGNITION FOR EVALUATING SOFTWARE

Assumed as a generalized criterion rather often may be an estimate of objective software performance, in other words the probability of job fulfilment determined by the expression [49] given below

$$E_1 = \int_{x_i \subset X} \rho(x_1, x_2, \ldots, x_n)\, dx_1\, dx_2 \ldots dx_n,$$

where $x_1, x_2, \ldots, x_n$ are the parameters; ρ is the probability density function; and x_i ($i = 1, 2, \ldots, n$) are elements of domain X. The optimal solution in this case may be found by maximizing an objective performance criterion or a performance-cost criterion, the other parameters of the system being subject to certain restrictions.

Thus, mathematical formulation of the problem gives

$$\max E(\mathrm{X}) \cap \min C(\mathrm{X}), \qquad (7.1)$$

$\mathrm{X} \subset \mathrm{X}$ assigned

where X is a set of particular performance criteria and parameters being considered.

Such an approach is valid when a generalized criterion can be represented as a function of particular criteria. As the function is usually unknown, it is represented as a series which in the simplest case is as follows

$$E(\mathrm{X}) = \sum_{i=1}^{n} x_i \alpha_i, \tag{7.2}$$

where E is the generalized performance criterion; x_i is a normalized value of the ith parameter; α_i is a weighting factor.

While using the formula (7.2) to evaluate software the following problems arise.

First, weighting factors should be evaluated by the formula

$$E(\mathrm{X}) = \sum_{i=1}^{n} \alpha_i' x_i + \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{ij}' x_i x_j, \tag{7.3}$$

where α_i', α_{ij}', $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, n$ are least-squares estimates with an assigned value of a generalized performance criterion. The linear function (7.2) can be used to evaluate the performance only when in the formula (7.3) the α_i variations are inessential and α_{ij}' are small. To choose reliably a generalized performance criterion, one must have a set of programs with the known quality criteria. This presents sufficient difficulties because, on the one hand, it seems impossible to select the most informative parameters among those characterizing the robot software and, on the other hand, the number of programs with the evaluated quality criteria is usually small. Therefore, the formula (7.3) cannot evaluate α_j' with a sufficient truth.

Second, great difficulties presents a simultaneous estimation of quantitative and qualitative parameters. In practice, however, they can be estimated and the performance evaluated with scarce initial information by groups of arguments in combination with pattern recognition [14, 16]. The problem of evaluating a program quality is defined as follows: each program is assigned a pattern determined by a set of features x_i which are parameters and particular quality criteria. Pattern class $\mathcal{H}_j$ corresponds to a generalized performance criterion. It should be noted that this generalized performance criterion within this approach can take only discrete values. The initial data to generate an algorithm of pattern recognition are a set of programs partitioned into groups according to the chosen quality criterion. If the *a priori* probabilities in each of the classes $p_k(\mathrm{X})$ are known, the problem is reduced to deriving the Bayes criterion of decision making. The information available determines hyperplanes dividing a hyperspace into classes to which any pro-

gram can be referred by using corresponding indices with a minimum of risk [14].

In case probability density of features characterizing each program cannot be assumed, the frequency of parameter in each class is estimated by the original sample. Then the domain of varying each parameter becomes discrete.

Class $\mathcal{H}_j$ determining the program quality by the feature vector and particular performance criteria, is calculated as follows:

$$\mathcal{H}_j = \underset{\mathcal{H}}{\operatorname{argmin}} \sum_{i=1}^{m} C(\mathcal{H}_i, \mathcal{H}_j)\, p(\mathcal{H}_i/\mathbf{X}), \tag{7.4}$$

where $C(\mathcal{H}_i, \mathcal{H}_j)$ characterizes losses while referring the software of class $\mathcal{H}_j$ to class $\mathcal{H}_i$, and m is the number of program complexes being considered. In this case, however, we need to know the estimates of frequencies $p(\mathbf{X}/\mathcal{H}_i)$ for all values of the vector $\mathbf{X}$, which seems impossible with a great number of parameters.

Therefore, to determine the generalized quality criterion the arguments are taken into account by groups, one group comprising parameters $x_k, x_l, \ldots, x_p$. In this case we can limit ourselves to a set of parameters defined by the expression [16]

$$\mathcal{H}_j = \underset{\mathcal{H}}{\operatorname{argmin}}\ C(\mathcal{H}_j/x_k, x_l, \ldots, x_p), \tag{7.5}$$

where $x_k, x_l, \ldots, x_p$ are the chosen set of parameters and

$$C(\mathcal{H}_j/x_k, x_l, \ldots, x_p) = \sum_{i=1}^{m} C(\mathcal{H}_i, \mathcal{H}_j)\, p(\mathcal{H}_j/x_k, x_l, \ldots, x_p)$$

is the risk of referring an object characterized by features $x_k, x_l, \ldots, x_p$ to class $\mathcal{H}_j$. The number of groups with a fixed number of parameters in the group l and with total of n features is determined by C_n^l.

What group is to be used to evaluate the performance is determined by introducing a check sample. The original sample is subdivided into two samples: learning $\mathbf{X}_l$ and check $\mathbf{X}_{ch}$. The former estimates the quantities $p(\mathcal{H}_j/x_k, x_l, \ldots, x_p)$ and the latter determines the groups which ensure the most precise division into classes. To choose a group, matrix $\mathbf{F}$ is formed which in the simplest case (two features in a group) can be represented as follows:

$$f_{l,v} = \begin{cases} \dfrac{1}{s} \displaystyle\sum_{i=1}^{s} \left| \mathcal{H}_j(x_l, x_{v+1}) - \underset{\substack{\mathcal{H} \\ v \geqslant l}}{\operatorname{argmin}} C(\mathcal{H}_j/x_l^i, x_{v+1}^i) \right| \\ 0;\ v < l \end{cases}$$

where s is the number of objects in the learning sample.

After the group providing the highest accuracy has been determined the algorithm generation could be considered completed. However, the number of parameters by which the software quality is evaluated would include only the features of this group. Herein we should like to note that the number of features in the group does not exceed three or four since even the learning sample with four grades and three features requires 64 objects, i.e. 64 programs with the known values of the generalized quality criterion. Such a number of objects is in general unknown. Therefore while sampling one group we usually limit ourselves to three or four parameters which are not always the most informative ones. In a method of group selection (7.5) the number of groups is determined by an exhaustive search with taking into account the evaluation precision. After f groups are selected an average error ε is evaluated as

$$\varepsilon = \frac{\sum\limits_{x_k, x_l, \ldots, x_p \subset X_{ch}} \sum\limits_{i=1}^{s} |\mathcal{H}_j(x_k, x_l, \ldots, x_p) - \operatorname{argmin}_{\mathcal{H}} C(\mathcal{H}_j/x_k, x_l, \ldots, x_p)|}{qS}, \tag{7.7}$$

where q is the number of elements in the check sample, the summing-up being carried out only with respect to the objects included into the check sample.

This does not complete synthesizing an algorithm of the generalized criterion evaluation. The class values calculated from the expressions (7.5) and (7.6), determine generalized features which are assigned to each of the selected groups. These secondary features are combined and the next step of the algorithm determined by the expressions (7.4)-(7.7) is performed. The criterion is formed until the precision $\xi = 1 - \varepsilon$ (7.7) stops increasing or reaches a maximum value equal to 1. If $\xi < \xi_{max}$, the classes are assigned incorrectly or the space of parameters characterizing the software is incomplete. The process of forming a generalized quality criterion employs a decision tree. The number of the selected groups of features gradually decreases and at the last step only one group remains. This group includes all the features determined at the previous steps.

To increase the rate of algorithm convergence, the matrix of losses C, determined as $C_{ij} = |i - j|$, is transformed at each step of the algorithm. For this purpose alongside with a total precision also calculated is the precision of determining the jth class

$$\xi_j = 1 - \frac{\sum\limits_{x_k x_l, \ldots, x_p \subset X_{ch}} |\mathcal{H}_j - \operatorname{argmin}_{\mathcal{H}} C(\mathcal{H}_j/x_k, x_l, \ldots, x_p)|}{q_j^m}, \tag{7.8}$$

where $q = \sum\limits_{i=1}^{m} q_i$, and the elements of the loss matrix at the rth step

of the algorithm are calculated as

$$C_{ij}^{r} = C_{ij}^{r-1}\xi_j^{r-1}, \quad i = 1, 2, \ldots, m, \quad j = 1, 2, \ldots, m. \tag{7.9}$$

The method of taking arguments into account by groups forms an algorithm of evaluating a generalized quality criterion with initial information insufficient for computing all the values of $p(\mathrm{X}/\mathscr{H}_l)$ for a complete vector of parameters X. Herein the evaluation precision is maximum for the initial available information.

The quality criterion for the program being analyzed can be calculated by including a corresponding pattern to the learning and check samples of each class and by reconstructing the criterion. The class for which the recognition of a corresponding pattern by the selected features will give the highest precision will determine the quality criterion. This method, however, is sufficiently labor-consuming though it allows the procedure of calculating the quality criterion to be corrected each time. Therefore in practice to evaluate the program quality it is advisable to use the expression (7.5) in which $p(\mathscr{H}_j/x_k, x_l, \ldots, x_p)$ is substituted by its evaluation found in calculating the quality criterion. In the general case it is as follows:

$$\prod_{x_k, x_l, \ldots, x_p \subset \mathrm{X}_{\mathrm{sel}}} p(x_k, x_l, \ldots, x_p/\mathscr{H}_j)_{r_{k,l,\ldots,p}}, \tag{7.10}$$

where $\mathrm{X}_{\mathrm{sel}}$ is a set of the selected groups of parameters.

The machine time of calculating the software quality essentially depends on a great number of parameters, and thus can be considerably decreased by reducing their number.

The most informative parameters are those which are included to the generalized quality criterion evaluation with a great weight. The number of the most informative parameters is determined by the number of iterative steps required to generate an algorithm of evaluating the quality criterion. The maximum number of parameters is equal to 2^k, where k is the number of iterations.

The evaluation precision essentially depends on the number of elements in the group and the number of groups taken account of at each step of the algorithm. However, an increase in the number of features in the group with an assigned number of classified objects reduces the number of nonzero elements in matrices determining the conditional probabilities of occurrence of parameter combinations in each class. Therefore an increase in the group volume increases the precision and decreases the reliability of results. The latter can be increased by introducing learning over all objects and by evaluating the precision ξ by the group selected from them. As the experiments have shown, the coefficient of filling in a matrix of conditional probabilities increases half as much without

causing any essential loss in the results precision. Such algorithms can be used to evaluate the most informative parameters and to determine a generalized quality criterion of the robot software.

7.3 GAME THEORY ANALYSIS OF SOFTWARE PERFORMANCE

The software system of a robot is designed to work for an extended time. Therefore it should be devised with due account for possible variations in the robotic complex operation. The program performance can also be evaluated by the game methods. Matrix games analyse the software with reference to the worst operating conditions. In a two-person game considered here, the strategy vector of the first player $\mathbf{y}^1$ determines the probability of using each software complex, and that of the second player $\mathbf{y}^2$ determines the probability of each strategy realization. The solution of the matrix game is as follows:

$$\mathbf{y}^{1T}\mathbf{A}\mathbf{y}^{20} \leqslant \mathbf{y}^{10T}\mathbf{A}\mathbf{y}^{20} \leqslant \mathbf{y}^{10T}\mathbf{A}\mathbf{y}^{2}, \tag{7.11}$$

where a_{ij} characterizes the quality of the ith software complex under the conditions that the jth situation is realized; $\mathbf{y}^{10}$ determines a set of program complexes effective with reference to the worst situation accomplished.

To choose a set of most effective programs in one criterion, generalized or particular, in case of realizing the worst conditions in another, recourse is had to bimatrix game

$$\mathbf{y}^{10T}\mathbf{A}^1\mathbf{y}^{20} \geqslant \mathbf{y}^{1T}\mathbf{A}^1\mathbf{y}^{20}, \tag{7.12}$$

$$\mathbf{y}^{10}\mathbf{A}^2\mathbf{y}^{20} \geqslant \mathbf{y}^{10T}\mathbf{A}^2\mathbf{y}^{2}.$$

After reducing game problems to a system of inequalities [40]

$$\begin{gathered}(C\mathbf{E}^2 - \mathbf{A}^{2T})\,\mathbf{y}^{1*} \geqslant \mathbf{l}^2;\ \mathbf{y}^{1*} \geqslant 0,\\ (\mathbf{y}^{2*},\ ((C\mathbf{E}^2 - \mathbf{A}^{2T})\,\mathbf{y}^{1*} - \mathbf{l}^2)) = 0,\\ (C\mathbf{E}^1 - \mathbf{A}^1)\,\mathbf{y}^{2*} \geqslant \mathbf{l}^1;\ \mathbf{y}^{2*} \geqslant 0,\\ (\mathbf{y}^{1*},\ ((C\mathbf{E}^1 - \mathbf{A}^1)\,\mathbf{y}^{2*} - \mathbf{l}^1)) = 0,\end{gathered} \tag{7.13}$$

they are solved by a Lemke-Howson algorithm. We would like to note that unlike (7.11) the conditions (7.12) cannot compute strategies insuring a preplanned result. The variation of probabilities of realizing possible situations of the software complex operation may reduce the performance of the programs by satisfying the relations

$$\alpha' = \mathbf{y}^{10T}\mathbf{A}^1\mathbf{y}^2 \leqslant \mathbf{y}^{10T}\mathbf{A}^1\mathbf{y}^{20} = \alpha^{10},$$

$$\alpha' = \mathbf{y}^{10T}\mathbf{A}^1\mathbf{y}^2 \leqslant \mathbf{y}^{1T}\mathbf{A}^1\mathbf{y}^2 = \alpha,$$

where $\mathbf{y}^{1T}$ and $\mathbf{y}^2$ do not form an equilibrium point.

To guarantee the program quality the following equilibrium conditions should be fulfilled

$$\mathbf{y}^{10\mathrm{T}}\mathbf{A}^1\mathbf{y}^{20} \geqslant \mathbf{y}^{10\mathrm{T}}\mathbf{A}^2\mathbf{y}^2,$$
$$\mathbf{y}^{10\mathrm{T}}\mathbf{A}^2\mathbf{y}^{20} \geqslant \mathbf{y}^{1\mathrm{T}}\mathbf{A}^2\mathbf{y}^{20}. \tag{7.14}$$

A number of enterprises engaged in producing separate elements take part in developing the software, and they are interested in increasing the performance criterion with reference to their production. In the general case the performance criterion of each player depends on the action of players, i.e. $(a^i_{ij} = f^i(y)$. The optimal strategy is selected by solving a polymatrix game the equilibrium strategies for which are determined by the following system of equations:

$$\sum_{i_1=1}^{n_1} \sum_{i_N=1}^{n_N} a^i_{i_1, \ldots, i_N} y^1_{i_1} \ldots y^N_{i_N} \geqslant \sum_{i=1}^{n_1} \ldots \sum_{i_N=1}^{n_N} a^i_{i_1},$$
$$\ldots, \; i_N y^{1^\circ}_{i,1} \qquad y^{(i-1)^\circ}_{i_{l-1}} y^{(i+1)^\circ}_{i_{l+1}} \qquad y^{N^\circ}_{i\mathrm{N}},$$
$$y^i_j \geqslant 0; \; j = 1, 2, \ldots n_i; \; \sum_{j=1}^{n_i} y^i_j = 1. \tag{7.15}$$

The game may be reduced to a nonlinear programming problem minimize

$$\sum_{i=1}^{N} \mathbf{A}^i\mathbf{y}^{i0}, \ldots \mathbf{y}^{N^\circ} - \sum_{i=1}^{N} p_i$$

subject to

$$\mathbf{A}^i\mathbf{y}^{1^\circ}, \quad ., \; \mathbf{y}^{N^\circ} \leqslant p_i.$$

The computation of equilibrium points, however, becomes quite complicated.

If the win of each player is determined by one more strategy the equations (7.15) will take the form

$$\mathbf{y}^{o\mathrm{T}i}\mathbf{A}^i\mathbf{y}^{oj} \geqslant \mathbf{y}^{\mathrm{T}i}\mathbf{A}^i\mathbf{y}^{oj}, \; I = 1, 2, \ldots, N,$$
$$j = F(i), \; \underset{i}{UF}(i) = I, \; F_{(i_k)} \neq F(i_l), \; k \neq l. \tag{7.16}$$

The pure strategy, say of the jth player, is taken as the initial value of the game. The initial value of the ith strategy of the jth player is determined by the vector $\mathbf{y}^{io}$

$$\mathbf{y}^{jo}_r = 0 \text{ for } r = r^*, \qquad \mathbf{y}^{jo}_{r^*} = 1/b^{F(i)}_{r^*k^*}, \tag{7.17}$$

where $r^* = \underset{r}{\operatorname{argmin}} (b^{F(i)}_{rk^*})$, and $k^* = \underset{k}{\operatorname{argmin}} (b^i_{k_1})$.

After the equilibrium conditions (7.16) are verified

$(\mathbf{e}_j^i,\ \mathbf{y}^i) \geqslant 0,\ (b^{F(i)},\ \mathbf{y}^i) \geqslant 1,\ j = 1, \ldots, m,$

each matrix $\mathbf{B}^i$ is transformed as follows

$\mathbf{B}^i = \mathbf{A}^i - \alpha \mathbf{E}^i,$

where $\alpha > \max\{b_{ij}^1, \ldots, b_{ij}^N\}$.

Each vector $\mathbf{y}_j^{io}$ defines a set

$$p^i(\mathbf{y}^i) = \{\mathbf{l}_{j1}^i,\ \mathbf{b}_k^{F(i)} \mid (\mathbf{l}_j^i,\ \mathbf{y}^i) = 0,\quad (\mathbf{b}_k^{F(i)},\ \mathbf{y}^i) - 1 = 0\}. \tag{7.18}$$

In the equilibrium point the following equilibrium conditions should be met

$$\mathbf{M}(\mathbf{y}_F) = \mathbf{M}(\mathbf{y}_{f_1}^1, \ldots, \mathbf{y}_{f_N}^N) = \{\mathbf{l}_f^i,\ i = 1, \ldots, N \mid \mathbf{l}_f^i \subset \mathbf{M}(\mathbf{y}_F), \text{ if}$$

$$\mathbf{l}_f^i \subset p^i(\mathbf{y}_{fi}) \cup b_f^i \subset p^{\mathrm{T}}(\mathbf{y}_{fi}^{\mathrm{T}})\}. \tag{7.19}$$

The matrix $\mathbf{B}^i$ being transformed, $p^i(\mathbf{y}_{fk})$ is changed in accordance with the expression

$$p^i(\mathbf{y}_f^i) = \{p^i(\mathbf{y}^{io}) \cup \mathbf{b}_{Nf}^{\mathrm{T}} \mid \mathbf{l}_f^i\}. \tag{7.20}$$

If the equilibrium conditions are satisfied the strategy vectors are determined as

$$\mathbf{y}_f^{i_k} = \mathbf{y}_f^{i_{k-1}} + \lambda_f^i \mathbf{l}_f^i,$$

where

$$\begin{aligned} \lambda_f^i &= \min_{b_{fi}<0;\ l_{jr}^i<0} \left(-\frac{(\mathbf{b}_f^i,\ \mathbf{y}^t - 1)}{b_{fi}},\ -\frac{y_r^t}{l_{jr}} \right), \\ \lambda_r &= \min_{b_{fj}>0;\ l_{jr}^i>0} \left(-\frac{(\mathbf{b}_f^i,\ \mathbf{y}^t - 1)}{b_{fi}},\ -\frac{y_r^t}{l_{jr}} \right). \end{aligned} \tag{7.21}$$

The computation is iterated in $k \leqslant n_i$ steps, where

$$n_i^* = \min_{i=1, \ldots, N} (n_i).$$

The solution of the game by using the equilibrium conditions (7.16), as in bimatrix games, does not guarantee a win outcome for each of the players. It could be insured only by satisfying the equilibrium conditions

$$\begin{aligned} &\mathbf{y}^{oTi} \mathbf{A}^i \mathbf{y}^{oj} \geqslant \mathbf{y}^{oTi} \mathbf{A}^i \mathbf{y}^j;\ i \subset I, \\ &(\mathbf{y}^i,\ \mathbf{l}^i) = 1;\ \mathbf{y}^i \geqslant 0;\ j = F(i), \\ &\bigcup_i F(i) = I;\ F(i_k) \neq F(i_l),\ k \neq l. \end{aligned} \tag{7.22}$$

The algorithm of the equilibrium point does not change [see (7.18)-(7.19)].

To make a preliminary estimate of performance, recourse is had to an antagonistic game

$$\mathbf{A}^i_{Ji} y^{io} \leqslant \alpha^i; \quad J^i = (1, \ldots, i-1, i+1, \ldots, N), \qquad \mathbf{y}^{io} \subset \mathbf{y}^i, \quad i = 1, \ldots, N, \tag{7.23}$$

in which players $F(i)$ form a coalition against i.

If pure strategies satisfying the relations exist

$$a^i_{j_1, \ldots, j_{i-1}, k, j_{i+1}, \ldots, j_N} \leqslant \sum_{j \subset J^i_n} a^i_{j_1, \ldots, j_{i-1}, j_{i+1}, \ldots, j_N}, \tag{7.24}$$

where $J^i_n \cup J^i_e = J^i$, the corresponding sets J^i_n determine nondominated strategy sets. The game values can be found by solving corresponding systems of linear equations. The convergence of the algorithm of solving matrix and polymatrix games essentially depends on the initial conditions. If $y^0 \subset J_e$ then, as

$$\left| \frac{y^i_p}{l^i_{ip}} \right| \leqslant \left| \frac{\sum_{j=1}^{n_i} y^i_j b_{ij} - 1}{b^i_{jp} - b^i_{kp} \frac{b_{il}}{b_{kl}}} \right|, \tag{7.25}$$

the algorithm closes and the solution is not found.

If the generalized performance criterion assumes only two values the analysis of the relation (7.24) reduces to verifying the domination with reference to each of the pure strategies separately, which considerably increases the rate of algorithm convergence by reducing the volume of calculations.

Game methods check how precisely the generalized quality criterion has been determined by the methods described in Section 7.1. To this end use is made of a reduced space of parameters, which is the same for each of the operating conditions being considered. Program quality estimates with reference to the reduced space of parameters, are used to construct $\mathbf{A}^0$ matrix. The difference between the game values of the initial matrix $\mathbf{A}^1$, where a^1_{ij} is a quality estimate of the ith program under jth conditions, determines precision ξ in the form

$$\xi = 1 - \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} (y^{10}_i a^1_{ij} y^{20}_j - y^{10'}_i a^0_{ij} y^{20'}_j). \tag{7.26}$$

Thus, the application of the game theory to select optimal robot software allows the decision maker not only to determine its compo-

sition with due account of operating conditions but also to find how precisely the performance has been estimated by pattern recognition.

List of Symbols

E	generalized performance criterion
x_i	parameters characterizing the software quality
$\mathbf{X}$	feature vector
$C(\mathscr{H}_i, \mathscr{H}_j)$	losses in referring the software of class $\mathscr{H}_j$ to class $\mathscr{H}_i$
$p(\mathbf{X}/\mathscr{H}_i)$	estimate of frequencies of features $x_k, x_l, \ldots, x_p$ in class $\mathscr{H}_i$
$\mathbf{A}$	payoff matrix
$\mathbf{y}^1$	strategy vector determining the probability of using each software complex
$\mathbf{y}^2$	strategy vector determining the probability of each strategy realization

8 Semiautomatic Robot Control Systems

A. S. Yushchenko, Cand. Sc. (Tech.)

8.1 GENERAL

At present it seems unexpedient to completely exclude man from controlling robotic systems, especially those with radioactive, space, and underwater applications. As the case may be, unpredictable situations would require active participation of the

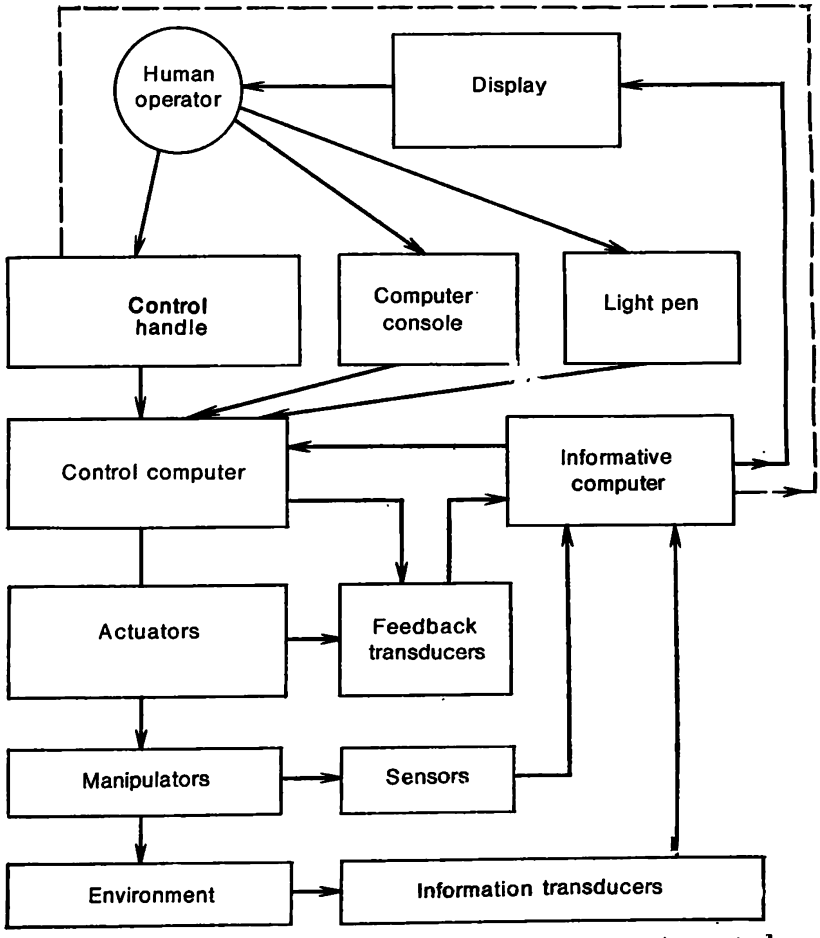


Fig. 8.1 Block diagram of a semiautomatic control

human operator. In master-slave manipulator systems [23] a man operating the manipulator has to perform all motions by his hand, the physical and psychical strain being especially great while executing monotonous operations. Semiautomatic systems, however, considerably facilitate the work of the operator, who is relieved of controlling the manipulator joints and has to control only hand motions by means of many-degrees-of-freedom control devices. Corresponding signals to the actuators are generated by microprocessor. Repeated operations can be easily programmed and are executed automatically under the operator's supervision. A number of operations, say, moving the hand to the required position, tool changing, etc., can be assigned by the operator through instructions. To increase the level of automation or to facilitate the work of the human operator semiautomatic systems employ sensitizing means. Figure 8.1 portrays a block diagram of a semiautomatic control system. The instructions for executing operations in an automatic mode are assigned via a microprocessor control console or by means of a light pen. In the main mode of semiautomatic control the operator assigns the motion of the manipulator hand by means of many-degrees-of-freedom control handle. The microprocessor generates signals to the actuators using force- and torque-feedback. The operator supervises the execution of operations with the help of devices displaying the manipulator position and environment.

8.2 SEMIAUTOMATIC CONTROL ALGORITHMS

In this paper semiautomatic control algorithms are means of forming control signals by the robot actuators depending on the control signals assigned by the human operator. At present to synthesize control algorithms two approaches are used. The first one is based on the kinematic relationships tying the manipulator hand coordinates, its velocities and accelerations with the generalized coordinates (with relative displacements in the joints). Hereafter we will refer to such control algorithms as kinematic. The second approach considers its dynamics and consists in forming forces governing a required motion.

We shall primarily discuss kinematic control algorithms. Most widespread is the method of rate control. A vector of angular and linear velocities of the manipulator hand is formed proportionally to the control handle deflections. Let $\mathbf{x}$ be a 6×1 vector determining the position and angular orientation of the manipulator hand; $\dot{\mathbf{x}}$ — a 6×1 velocity vector of the hand center of mass (together with the workpiece) and its angular velocity; $\mathbf{q}$ — an $N \times 1$ vector of the manipulator generalized coordinates; $\mathbf{J}(\mathbf{q})$ the Jacobian matrix, $\mathbf{x} = f(\mathbf{q})$; $\mathbf{U}$ —a 6×1 vector corresponding to control signals

(deflections of the control handle). Then the problem of forming control signals is reduced to solving the following system of equations

$$\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\,\dot{\mathbf{q}} \tag{8.1}$$

with respect to the unknown vector $\dot{\mathbf{q}}$ at $\dot{\mathbf{x}} = k\mathbf{u}$, where k is the scale coefficient.

The matrix $\mathbf{J}(q) = \left[\frac{\partial f_i}{\partial q_j}\right]$, $i = 1, \ldots, 6$, $j = 1, \ldots, N$

is determined from the formula

$$\mathbf{J} = \begin{vmatrix} (\mathbf{e}_0 \times \mathbf{R}_{0N})\,\sigma_1 + \mathbf{h}_0(1-\sigma_1) \ldots (\mathbf{e}_{N-1} \times \mathbf{R}_{N-1,\,N})\,\sigma_N + \mathbf{h}_{N-1}(1-\sigma_N) \\ \mathbf{e}_0\sigma_1 \qquad\qquad\qquad\qquad \mathbf{e}_{N-1}\sigma_N \end{vmatrix}, \tag{8.2}$$

where $\mathbf{R}_{1N}$ are position vectors traced from the center of the ith kinematic pair to the hand center of mass ($\sigma_i = 1$, in the case of a revolute pair and $\sigma_i = 0$ in the case of a prismatic pair); $\mathbf{e}_i$, $\mathbf{h}_i$ are unit vectors of the axes of the revolute and prismatic pairs, respectively. If the matrix $\mathbf{J}$ is square and nonsingular, the equation (8.1) gives us the vector $\dot{\mathbf{q}}$ as

$$\dot{\mathbf{q}} = \mathbf{J}^{-1}(\mathbf{q})\,\dot{\mathbf{x}}. \tag{8.3}$$

The matrix $\mathbf{J}$ may have a maximum rank equal to six while the number N of joints in the mechanism is greater than six. In this case the equation (8.1) has a plurality of solutions. To select one of them an additional conditions should be introduced, e.g. we may select a solution which minimizes a quadratic functional

$$\mathbf{J}_1 = \frac{1}{2}\,\dot{\mathbf{q}}^{\mathrm{T}}\mathbf{L}\dot{\mathbf{q}}, \tag{8.4}$$

where $\mathbf{L}$ is nonsingular symmetric $N \times N$ matrix. This functional may be considered as the kinetic energy of the system. The solution of such a variational problem is as follows [50]:

$$\dot{\mathbf{q}} = \mathbf{L}^{-1}\mathbf{J}^{\mathrm{T}}(\mathbf{J}\mathbf{L}^{-1}\mathbf{J}^{\mathrm{T}})^{-1}\,\dot{\mathbf{x}}. \tag{8.5}$$

The vector $\dot{\mathbf{q}}$ cannot be found from the formula (8.1) also when the rank of the matrix $\mathbf{J}$ is smaller than its maximum value (the matrix $\mathbf{J}$ becomes singular). This is possible while moving along an arbitrary path; the points $\mathbf{q}_0$ where the matrix singularizes, can be found from the condition

$$\det \mathbf{J}(\mathbf{q}_0) = 0. \tag{8.6}$$

With constraints imposed on the motions in the manipulator joints

$$q_{i\,\min} \leqslant q_i \leqslant q_{i\,\max}, \tag{8.7}$$

the matrix **J** also singularizes when $q_i = q_{i\,\min}$ or $q_i = q_{i\,\max}$, because further motion of the mechanism proceeds as though it has minus one joint. In this case the equations (8.1) can approximately be solved by linear programming [43]. Introduced in this case are new variables assuming only positive values

$$\dot{q}_i = \dot{q}_i^{+} - \dot{q}_i^{-}, \quad \dot{q}_i^{+} \geqslant 0; \quad \dot{q}_i^{-} \geqslant 0, \; i = 1, 2, \ldots, N,$$

a $(2N+1) \times 1$ vector

$$\tilde{\dot{\mathbf{q}}} = [\dot{q}_1^{+} \ldots \dot{q}_N^{+} \dot{q}_1^{-} \ldots \dot{q}_N^{-} q_{N+1}], \; q_{N+1} \geqslant 0$$

and a linear function

$$I_2 = q_{N+1} + \sum_{i=1}^{N} \gamma_i (\dot{q}_i^{+} + \dot{q}_i^{-}), \quad \gamma_i \geqslant 0, \quad i = 1, \ldots, N. \tag{8.8}$$

Let us formulate the problem of the linear function I_2 minimization under the following constraints

$$\sum_{j=1}^{N} a_{ij} \dot{q}_j - \dot{x}_i \leqslant q_{N+1}; \; i = 1, \ldots, 6,$$

i.e.

$$-q_{N+1} \leqslant \sum_{j=1}^{N} a_{ij} [\dot{q}_j^{+} - \dot{q}_j^{-}] - \dot{x}_i \leqslant q_{N+1}, \tag{8.9}$$

where a_{ij} are elements of the matrix $\mathbf{J}(\mathbf{q})$.

The constraints on $\dot{q}_i^{+}$, $\dot{q}_i^{-}$, and q_{N+1} are of the type $\geqslant 0$ and those on the relative motions in the manipulator joints with the account of the equality $q_i + \Delta q_i \approx q_i + \dot{q}_i^{+} - \dot{q}_i^{-}$ may be rewritten as

$$q_{i\,\min} - q_i \leqslant \dot{q}_i^{+} - \dot{q}_i^{-} \leqslant q_{i\,\max} - q_i. \tag{8.10}$$

With (8.9), (8.10) fulfilled, it is necessary to select a vector $\tilde{\dot{\mathbf{q}}}$ minimizing I_2. As q_{N+1} enters I_2 with a unit coefficient, minimizing I_2 results in minimizing the difference $\mathbf{J}\dot{\mathbf{q}} - \dot{\mathbf{x}}$ for each of the vector components. Minimizing the component $\sum_{i=1}^{N} \gamma_i (\dot{q}_i^{+} + \dot{q}_i^{-})$ provides such a selection of $\dot{q}_i^{+}$, $\dot{q}_i^{-}$ that the absolute value of $|\dot{q}_i| \leqslant \dot{q}_i^{+} + \dot{q}_i^{-}$ is minimal. The formulated problem of linear programming can be solved by means of the known numerical methods, the dual simplex method including. Solving a linear programming problem for

each step is greatly machine time consuming. Therefore it can be used only for robots automatically controlled by means of computers.

With semiautomatic control we need such method of approximate solving of the equations (8.1) that would enable us to perform real time computations. To this end we can use a regularization method suggested by A. N. Tikhonov [57]. The solution of the equations (8.1) is then replaced by minimizing a smoothing functional of the form

$$I_3 = \| J\dot{q} - \dot{x} \|^2 + \gamma (C^2(\dot{q}) + \lambda\Omega(\dot{q})), \tag{8.11}$$

where

$$C^2(\dot{q}) = \dot{q}^T \chi \dot{q}; \quad \Omega(\dot{q}) = (\dot{q} - \dot{q}_0)^T \omega (\dot{q} - \dot{q}_0);$$

$$\chi = \operatorname{diag}[\chi_1 \quad \chi_N], \quad \omega = \operatorname{diag}[\omega_1 \quad \omega_N];$$

$$\lambda, \ \gamma, \ \omega_j, \ \chi_j = \text{const},$$

$$\| x \| = \Big(\sum_{i=1}^{N} x_i^2\Big)^{1/2}.$$

Here $\dot{q}_0$ is the value of the vector $\dot{q}$ at the preceding step. The component $C^2(\dot{q})$ insures selection of solutions minimal in the norm $\big(\sum_{i=1}^{N} q_i^2 \chi_i\big)^{1/2}$ and the component $\Omega(\dot{q})$ enables us to select solutions which deviate minimally from the solutions obtained at the preceding step.

The functional being minimal (8.11), we shall find

$$\dot{q}_{\gamma, \lambda} = [J^T J + \gamma (\chi^T \chi + \lambda\omega)]^{-1} \times (J^T \dot{x} + \gamma\lambda\omega\dot{q}_0). \tag{8.12}$$

If an exact solution $\dot{q}^*$ of the equation (8.1) exists, the reduction in parameters γ, λ makes the approximate solution $\dot{q}_{\gamma, \lambda}$ converge to the exact one

$$\lim_{\gamma, \lambda \to 0} \| \dot{q}^* - \dot{q}_{\gamma, \lambda} \| = 0. \tag{8.13}$$

The formula (8.12) determines an algorithm of calculating control signals to manipulator joints depending on the assigned vector $\dot{x}$ and position of the joints determined by the vector q_0 on which in its turn the matrix $J = J(q_0)$ depends.

As distinguished from a linear programming problem, the solution (8.12) is explicitly defined. Therefore when the matrix $[J^T J + \gamma (\chi^T \chi + \lambda\omega)]^{-1}$ can be written analytically the regularization is

time saving. For more complicated problems, however, the matrix inversion becomes commensurable in time with the linear programming.

We shall note that the formulated problem can be interpreted as minimizing the regularizing functional $C^2(\dot{\mathbf{q}}) + \lambda\Omega(\dot{\mathbf{q}})$ over the set of solutions of the equation (8.1) obtained by Lagrange's method of undetermined multipliers. In particular the power N to change the potential energy of the manipulator can be regarded as $C(\dot{\mathbf{q}})$ and its kinetic energy as $\Omega(\dot{\mathbf{q}})$. When $N \leqslant 0$ we must assume $\chi = 0$. Then the solution of the problem of minimizing the functional (8.11) will result in selecting that solution out of a set of solutions of the system (8.1) which ensures minimum of the manipulator kinetic energy and power to overcome static moments in the joints caused by the limbs weight.

The algorithm of position control can be built on the basis of rate control algorithm. In this case the position and orientation of the hand should correspond to that of the control device. Supposing that $\delta\mathbf{q}$, $\delta\mathbf{x}$ are small vector deviations at the end of the calculation step as compared to the beginning of the step, we can rewrite (8.1) as

$$\delta\mathbf{x} = \mathbf{J}(\mathbf{q})\,\delta\mathbf{q}. \tag{8.14}$$

Assuming

$$\delta\mathbf{x} = k_0(\mathbf{u} - \mathbf{x}) = k_0(\mathbf{u} - f(\mathbf{q})), \tag{8.15}$$

the expression (8.14) can be used to find the corresponding vector deviation $\delta\mathbf{q}$. We shall note that as distinct from rate control in this case the manipulator should have position feedback actuators. In particular we can assume in the expression (8.15) $\mathbf{u} = \mathbf{x}_d$, where $\mathbf{x}_d$ is a vector of the desired hand position and orientation. Then the hand will automatically be moved to the required position.

Under real conditions the operator can face different problems: to move the manipulator hand to an assigned point and to position the hand in close vicinity of the working object. Therefore a semiautomatic rate control system should operate in any of the three modes described above, i.e. it must be a combined system.

The algorithm of controlling a combined system is visualized in Fig. 8.2. To select a required mode of operation, two logic variables η, γ are used. Each of them can assume two values 0 or 1. To move the manipulator to an assigned point $\mathbf{x}_d$, η should equal zero with the semiautomatic control or one with the automatic control. The variable γ equals zero for rate control and one for position control with small deflections of the control handle. The operator $\mathbf{B} = \tilde{\mathbf{J}}^{-1}(\mathbf{q})$ realizes approximate solving of the equation (8.1) with the measured value of the generalized coordinates.

In a semiautomatic mode a change between rate and position modes is to occur at the neutral position of the control handle ($\mathbf{u} = 0$). This assumption is valid while switching from a rate mode of control to a position one since the hand velocity in close vicinity of the workpiece is equal to zero. However, when changing the modes from

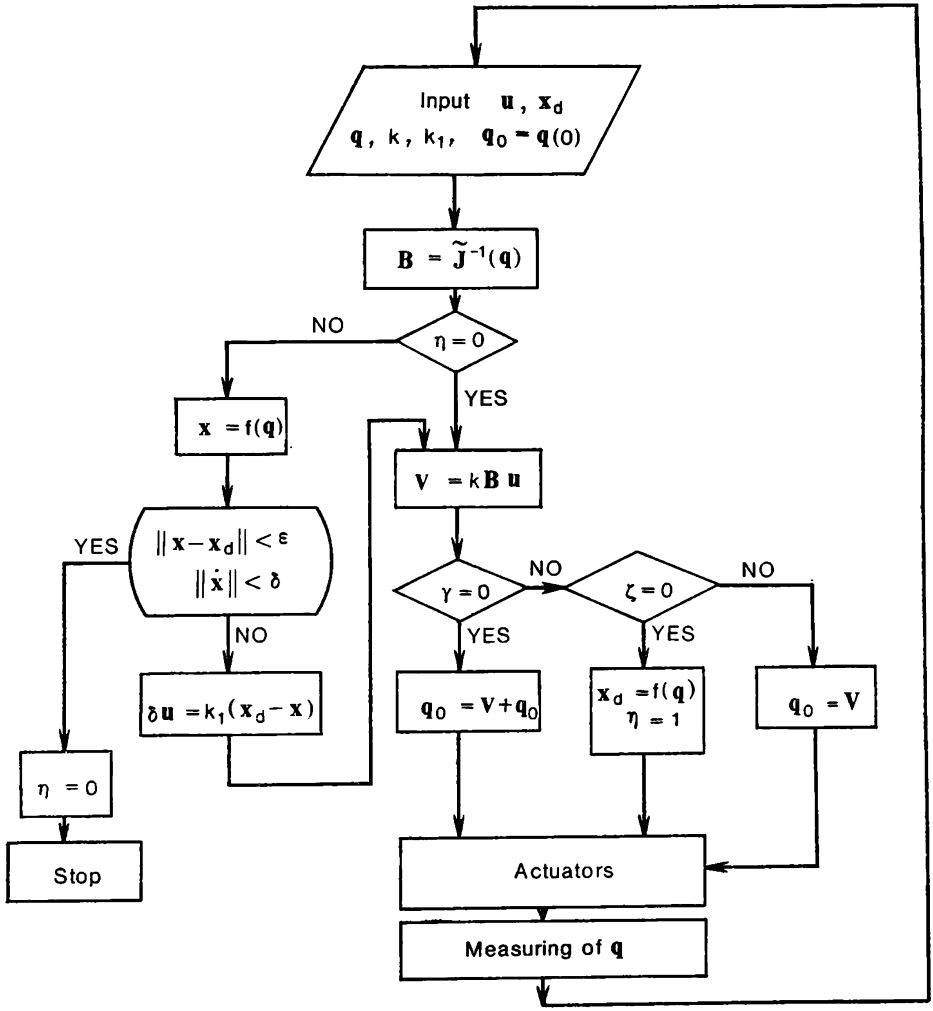


Fig. 8.2 Algorithm of combined rate control system

a position to a rate one transient responses may occur because the position of the control handle corresponding to a required hand location differs from a neutral one. To avoid transients the block diagram comprises one more logic variable ξ which becomes zero when in

position control the operator moves the control handle to a neutral position. Then, the system stabilizes with respect to the position the manipulator occupied at the moment of switching and rate control may be implemented.

Switching from one mode of operation to another in the combined system may be performed either through instructions from the human operator or automatically. In the latter case the logic of switching depends on the character of performed operations. For example, while executing jobs requiring care, automatic reswitching from a rate mode to a precise position mode of control may be accomplished to avoid the hand collision with a manipulated object.

The control device used by the human operator in a system of rate or position control incorporates force-feedback. In application to semiautomatic manipulator systems we can consider bilateral semiautomatic systems analogous to master-slave manipulator systems [24].

Force-feedback in position controlled systems facilitates the work of the human operator in those cases when considerable external forces are exerted on the manipulated object. At the same time force-feedback may worsen the system's dynamic performance and may lead *inter alia* to the occurrence of self-excited oscillations at the boundary of a free zone, i.e. when the hand touches the obstacle [44].

8.3 SYSTEM OF MANIPULATOR HAND SELF-DIRECTING CONTROL

One of the most frequently repeated operations of a manipulator is the hand motion to the vicinity of a workpiece. If the workpiece coordinates are known the task can be fulfilled in an automatic mode as described above. More often than not, the workpiece coordinates are not known in advance. Moreover, it may travel past the manipulator or the manipulator may move to track a workpiece, which essentially complicates the task fulfilment in a semiautomatic mode of operation.

Besides, limited time for task fulfilment and little information on relative positions of the hand and workpiece additionally complicate the task. To cope with these difficulties use is made of a control system which automatically moves the hand to the vicinity of a workpiece (self-direction).

For task implementation the robot is provided with an ultrasonic locator.

To realize control the manipulator should be designed so that hand translation and orientation proceed irrespective of each other. The last three joints of the kinematic chain which orient the hand

are used for searching and subsequent holding the workpiece in the locator equisignal zone. The rest N-3 joints of the manipulator form a "base mechanism" which moves the locator.

The hand self-directing control can be based on the proportional navigation method [17] which assumes the angular speed of sighting $\dot{\Psi}$ to be proportional to the angular speed of the hand absolute speed vector $\dot{\varphi}$, i.e.

$$\dot{\varphi} = k\dot{\Psi}. \tag{8.16}$$

With different proportionality factors k we shall have different types of self-directing control.

Vector $\dot{\Psi}$ is calculated from the formula

$$\dot{\Psi} = \sum_{i=1}^{N-3} \sigma_i \mathbf{e}_{i-1}\dot{q}_{i-1} + \tau_{0,\,N-3}\dot{\Psi}^{(N-3)}, \tag{8.17}$$

where $\dot{\Psi}^{(N-3)} = \mathbf{e}_{N-3}^{(N-3)}\dot{q}_{N-2} + \mathbf{e}_{N-2}^{(N-3)}\dot{q}_{N-1}$

is the vector $\dot{\Psi}$ in the coordinate system tied to the last link of the base mechanism; $\tau_{0,N-3}$ is an orthogonal matrix of transition from the last-link-tied frame to the manipulator-base-tied frame. Values q_i and $\dot{q}_i$ are measured by corresponding transducers. The expression (8.1) determines the hand velocity vector $\mathbf{V}(t)$ direction; the magnitude of this vector is dependent on the workpiece velocity and distance.

Now the base mechanism control can be furnished in a rate mode. Let us denote by $\mathbf{q} = [q_1, \ldots, q_{N-3}]^T$ the vector of the generalized coordinates of the base mechanism; by $\mathbf{J}_b$ the Jacobian matrix for the equations relating the coordinates of the center of intertia of the last link of the base mechanism to its generalized coordinates, defined as the first line of the partitioned matrix (8.2) written for the base mechanism. If, in particular, the matrix is 3×3, i.e. $N = 6$, three joints of the base mechanism are used, and $\det \mathbf{J}_b \neq 0$ we can write

$$\dot{\mathbf{q}} = \mathbf{J}_b^{-1}\mathbf{V}. \tag{8.18}$$

Since the hand velocity vector has already been determined, the expression (8.18) can give us control signals to the basic mechanism actuators. Fig. 8.3 visualizes a block diagram of the robot self-directing control system. Vector $\dot{\Psi}^{(N-3)}$ measured by the locator is recomputed by unit *2* of the calculator into vector $\dot{\psi}$ determined in the inertial frame by the formula (8.17); the generalized coordinates of the base mechanism and their derivatives enter the computation. Furtheron unit *3* calculates the vector $\dot{\varphi}$ by the formula (8.16);

integrating this vector we can obtain the vector φ determining an angular orientation of the vector V. Assigning the magnitude $|V|$ we can get the vector V (unit *4*) and using the formula (8.18) or the approximate relations of rate control discussed in Section 2 we can calculate $\dot{q}$ (in unit *5*).

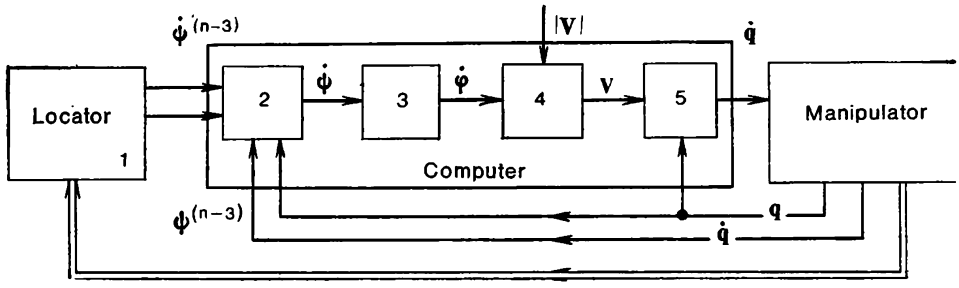


Fig. 8.3 Block diagram of robot self-directing control system

To illustrate we shall consider a computer-simulated self-directing control system for a mechanism having three revolute pairs

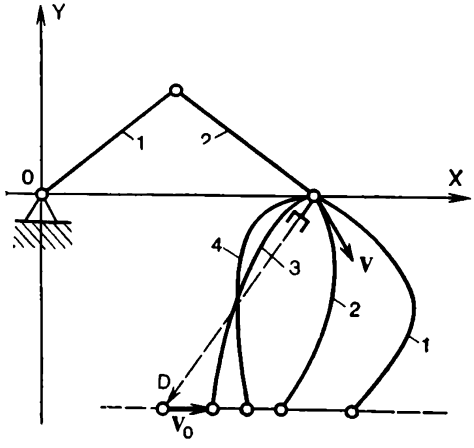


Fig. 8.4 Computer-simulated self-directing control system for a mechanism having three revolute pairs

with parallel axes (Fig. 8.4). The axis of the locator equisignal zone, as we shall assume, is always directed along the axis of the manipulator last link. Then

$$\dot{\Psi} = \sum_{i=1}^{3} \dot{q}_i \mathbf{e}_{i-1}.$$

The angular speed $\dot{\varphi}$ is determined from the equation (8.16); integrating $\dot{\varphi}$ we can find φ and calculate $V_x = |\mathbf{V}| \cos \varphi$, $V_y = |\mathbf{V}| \sin \varphi$. In the expression (8.18) $\mathbf{J}_b$ is a 2×2 matrix and its components are found from the formula

$$\mathbf{J}_b = [\mathbf{e}_0 \times \mathbf{R}_{03}\mathbf{e}_1 \times \mathbf{R}_{13}] \tag{8.19}$$

as

$$a_{11} = l_1 \sin q_1 + l_2 \sin (q_1 + q_2),$$

$$a_{12} = l_2 \sin (q_1 + q_2),$$

$$a_{21} = -l_1 \cos q_1 - l_2 \cos (q_1 + q_2),$$

$$a_{22} = -l_2 \cos (q_1 + q_2).$$

To describe completely the manipulator hand self-directing control system we should add to the equations (8.16-8.18) an equation defining the law of varying the distance vector $\mathbf{D}$

$$\dot{\mathbf{D}} = \mathbf{V}_0 - \mathbf{V}, \tag{8.20}$$

where $\mathbf{V}_0$ is the absolute velocity of a workpiece motion. Then the following system of nonlinear equations can be obtained

$$\begin{aligned} |\dot{\mathbf{D}}| &= V_0 \cos (\Psi - \varphi_0) - V \cos (\Psi - \varphi), \\ |\mathbf{D}|\,\dot{\Psi} &= V_0 \sin (\varphi_0 - \Psi) + V \sin (\Psi - \varphi). \end{aligned} \tag{8.21}$$

Figure 8.4 presents computer-simulated graphs of self-directing control at a constant speed magnitude (V=0.3 m/s for curves *1*, *2*; V = 0.4 m/s for curves *3*, *4*). The workpiece moves rectilinearly at a constant speed V_0 = 0.2 m/s. The increase in coefficient k makes the path more straight and time of hand control respectively decreases (k = 5 for curves *1*, *4*; k = 10 for curves *2*, *3*). Thus selecting the coefficient k we can achieve a required quality of control. Naturally the extreme value of the coefficient k is limited by the stability conditions of the control system.

8.4 DYNAMIC CONTROL OF MANIPULATORS

As the name implies, the dynamic control of manipulators involves control of forces and moments applied by the manipulator hand to a workpiece.

The manipulator motion together with a workpiece held in hand without letting go, can be described by the following equation

$$\mathbf{B}^{0T} [\mathbf{F}_e + \mathbf{F}\mathbf{w}^0 \mathbf{M}_e + \mathbf{M}_I]^T + \mu = 0, \tag{8.22}$$

where $\mathbf{F}_e$, $\mathbf{M}_e$ are external forces and moments acting on the manipulator links; $\mathbf{F}_I = -\mathbf{m}\mathbf{w}^0$ the block vector of the manipulator inertia main forces; $\mathbf{m} = \text{diag}\,[m_1\mathbf{I}_3 \ldots m_N\mathbf{I}_3]$ the partitioned matrix of the masses of links m_i; $\mathbf{I}_3$ the 3×3 identity matrix; $\mathbf{w}^0$ the block

vector of the accelerations of centers of inertia; $\mathbf{M}_I = [\mathbf{M}_I, \ldots, \mathbf{M}_{IN}]$ the block vector of the inertia main moments of the manipulator links; $\boldsymbol{\mu}$ the forces and moments developed by the actuators in the axes of kinematic pairs; $\mathbf{B}^{0T}$ the $6N$ N partitioned matrix described in [32]. In the equation (8.22), the mass of the workpiece with which the manipulator operates is referred to the Nth link. The inertia tensor of the Nth link $\mathbf{I}_N$ is determined together with the workpiece. We shall substitute the vector $\boldsymbol{\mu}$ in the equation (8.22) by the equivalent component $\mathbf{J}^T[\mathbf{F} \mid \mathbf{M}]^T$, where $[\mathbf{F} \mid \mathbf{M}]$ is the vector of forces and moments applied to the Nth link by the manipulator, and $\mathbf{J}$ is the Jacobian matrix for $\mathbf{x} = f(\mathbf{q})$, relating the vector $\mathbf{x}$ with the manipulator generalized coordinates $\mathbf{q}$ (see Section 2). Then the equation (8.22) will assume the form

$$\mathbf{B}^{0T}[\mathbf{F}_e + \mathbf{F}_I \mid \mathbf{M}_e + \mathbf{M}_I]^T + \mathbf{J}^T[\mathbf{F} \mid \mathbf{M}]^T = 0 \tag{8.23}$$

In particular, if the external forces are applied to a workpiece only we shall have

$$\mathbf{J}^T[\mathbf{F}_e + \mathbf{F} \mid \mathbf{M}_e + \mathbf{M}]^T + \mathbf{B}^{0T}[\mathbf{F}_I \mid \dot{\mathbf{M}}_I]^T = 0. \tag{8.24}$$

Let us assign the workpiece motion program by a set of equations

$$\Phi_\lambda(\mathbf{x}) = 0, \ \lambda = 1, \ldots, l. \tag{8.25}$$

Geometric constraints imposed on the workpiece motion will be represented by the following system of inequalities

$$\Phi_\varkappa(\mathbf{x}) \leqslant 0, \ \varkappa = 1, \ldots, m. \tag{8.26}$$

Then, the problem of dynamic control is generally formulated so as to select a vector of forces $\mathbf{F}$ and moments $\mathbf{M}$ from the equations (8.24) compatible with constraints (8.25), (8.26). The solution of this problem is to provide such a motion of the actuator which performs an assigned operation. In a semiautomatic control system a control vector of forces and moments $[\mathbf{F} \mid \mathbf{M}]$ is assigned by the human operator by applying proportional forces and moments $\mathbf{F}_0$, $\mathbf{M}_0$ to a control handle. To fulfil the motion to an assigned program the human operator must be well informed of the external forces and moments and also of links reaction. Therefore the control method implies displaying forces and moments on a control device.

The forces and moments in the manipulator joints are determined by the formula

$$\boldsymbol{\mu} = \mathbf{J}^T \mathscr{F}, \tag{8.27}$$

where $\mathscr{F}$ is a $[6 \times 1]$ vector $[\mathbf{F} \mid \mathbf{M}]^T$.

If the rank of the matrix $\mathbf{J}$ is equal to six then in statics, as it follows from the equation (8.24), the manipulator hand will be in equilibrium under the influence of the external and control forces.

A control algorithm based on the formula (8.27) is called the algorithm of force control.

To provide a stable operation of the system, the vector of control forces and moments is chosen as a function of the control vector $\mathbf{u}$, hand coordinates and their derivatives

$$\mathcal{F} = \Phi(\mathbf{x}, \dot{\mathbf{x}}, \mathbf{u}). \tag{8.28}$$

The law of motion depends on the type of this function. If, in particular, we shall take $\Phi(\mathbf{x}, \dot{\mathbf{x}}, \mathbf{u}) = k_0\mathbf{u} = k_0\delta\mathbf{x}_0$, where $\delta\mathbf{x}_0$ is the deflection of a control handle caused by the human operator we shall come to the formula (8.27). Suppose

$$\Phi(\mathbf{x}, \dot{\mathbf{x}}, \mathbf{u}) = k_0\delta\dot{\mathbf{x}}_0 - k_1\dot{\mathbf{x}}, \tag{8.29}$$

and $k_0 = k_1$, then with such a selection of control algorithm the deflection of a control handle is assigning a velocity vector $\dot{\mathbf{x}}_d$. So the force applied to the manipulator hand will be proportional to a velocity error $\dot{\mathbf{x}}_d - \dot{\mathbf{x}}$ and will be directed so as to decrease it. Thus, in a steady state the manipulator operation will be controlled in a rate mode. If the control mode is close to a static one, i.e. $\dot{\mathbf{x}} \approx 0$, the formula (8.29) still determines force control.

By selecting the vector $\mathcal{F}$ we can move the manipulator to a specified point $\mathbf{x}_d$. To this end we can assume

$$\Phi(\mathbf{x}, \dot{\mathbf{x}}, \mathbf{x}_d) = k_0(\mathbf{x}_d - \mathbf{x}) - k_1 \tag{8.30}$$

then at each moment of time applied to the hand will be a vector of forces and moments proportional to the position error (linear and angular) and directed so as to decrease it. The component $k_1\dot{\mathbf{x}}$ ensures a stable transient in the vicinity of a target point (for greater detail see Section 5).

The formulas (8.29) and (8.30) can be rewritten as

$$\mathcal{F} = k_0\mathbf{u} - k_1\dot{\mathbf{x}}, \tag{8.31}$$

where the vector $\mathbf{u}$ is equal either to $\delta\mathbf{x}_0$ or to $\mathbf{x}_d - \mathbf{x}$. This expression forms the basis for the algorithm of combined force control [55], visualized in Fig. 8.5. The modes of control are assigned by a logic variable η, with $\eta = 1$ the manipulator hand is automatically moved to the position $\mathbf{x}_d$, while $\eta = 0$ corresponds to a semiautomatic mode of control. When the human operator shifts the control handle to the neutral position the manipulator does not stop, as distinct from rate control (see Fig. 8.2), but moves with a speed it had at the moment of shifting. It is quite permissible since in force control

the operator often makes recourse to a pulse control when the control handle is in a neutral position while the hand moves at a constant speed. Whenever human operators are involved, blocking is needed to stop the manipulator in a position it had at the moment of unpredictable control handle release (to prevent an uncontrolled motion). For this purpose one more logic variable ξ is introduced.

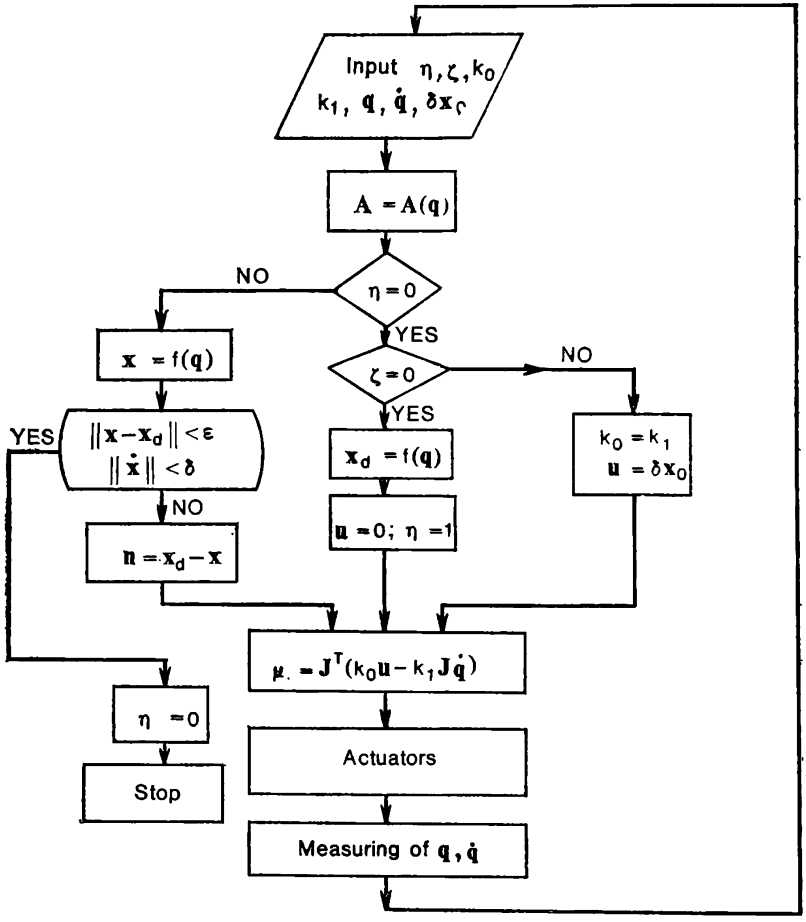


Fig. 8.5 Algorithm of combined force control

In the semiautomatic control $\xi = 1$. If the operator releases the control handle, ξ becomes zero and the system stabilizes with respect to the position the manipulator occupied at the moment the control was stopped. In the case of emergency stop, ξ is also assumed equal to zero. Obviously the control handle must comprise switches and sensors to judge whether the operator holds the stick or not.

In a semiautomatic mode $\mathbf{u} = \delta\mathbf{x}_0$. From current values of the manipulator generalized coordinates and their derivatives, control moments are computed by the formula

$$\boldsymbol{\mu} = \mathbf{J}^{\mathrm{T}}(\mathbf{q})\,(k_0\mathbf{u} - k_1\mathbf{J}(\mathbf{q})\,\dot{\mathbf{q}}), \tag{8.32}$$

and vector $\mathbf{x} = f(\mathbf{q})$ determining the hand orientation is found.

The algorithm of force control requires the use of moment motors or actuators with torque-feedback.

We illustrate by Fig. 8.6 the computer-simulated motion of a three-link manipulator in a vertical plane. The manipulator links are rods with central inertia moments equal to $I_1 = 0.0186$, $I_2 = 0.089$, $I_3 = 0.0019$ kg m^2, the length $l_1 = 0.3$, $l_2 = 0.26$, $l_3 = 0.2$ m, and mass $m_1 = 2.3$, $m_2 = 1.5$, $m_3 = 0.6$ kg. Fig. 8.6a visualizes motion corresponding to position control with

$$\mathcal{F} = \mathbf{F} = F_0\,(\mathbf{x}_{\mathrm{d}} - \mathbf{x})/\|\,\mathbf{x}_{\mathrm{d}} - \mathbf{x}_0\,\| - k_1\dot{\mathbf{x}},\ k_1 = 3,\ F_0 = 2\mathrm{N},$$

Fig. 8.6b that corresponding to rate control

$$\mathbf{u} = k_1\,(\dot{\mathbf{x}}_{\mathrm{d}} - \dot{\mathbf{x}}),\ |\dot{\mathbf{x}}_{\mathrm{d}}| = 0.14\ \mathrm{m/s},\ k_1 = 2,$$

and Fig. 8.6c shows motion with zero speed at the assigned point, i.e.

$$\dot{\mathbf{x}}_{\mathrm{d}} = \dot{\mathbf{x}}_0\,(\mathbf{x}_{\mathrm{d}} - \mathbf{x}_0)/\|\,\mathbf{x}_{\mathrm{d}} - \mathbf{x}_0\,\|,$$

$|\dot{\mathbf{x}}_0| = 0.2$ m/s, $k_1 = 2$.

As the simulation results have shown, the algorithms of automatic force control insure a required law of control but at the sacrifice of transients quality as compared with algorithms of kinematic control described in Section 2. This is due to the effect of components in the dynamics equation (8.23) taking account of the inertia properties of the manipulator proper. They also worsen the system's operation in a semiautomatic mode since the inertia forces and torques are not measured by sensors mounted in the manipulator wrists but are only judged of by the human operator who watches motion deviations caused by them. Therefore it is advisable to compensate dynamically the robot inertia forces and torques.

Let us discuss this suggestion in detail. We shall write the equation of motion (8.23) supposing that the gravitational forces are compensated by a balance system in the form

$$\boldsymbol{\mu} + \mathbf{J}^{\mathrm{T}}(\mathcal{F}_{\mathrm{e}} + \mathcal{F}_{IN}) + \mathbf{B}^{0\mathrm{T}}\mathcal{F}_{\mathrm{I}}' = 0,$$

where $\mathcal{F}_{\mathrm{e}} = [\mathbf{F}_{\mathrm{e}} \vdots \mathbf{M}_{\mathrm{e}}]^{\mathrm{T}}$; $\mathcal{F}_{IN} = [\mathbf{F}_{IN} \vdots \mathbf{M}_{IN}]^{\mathrm{T}}$

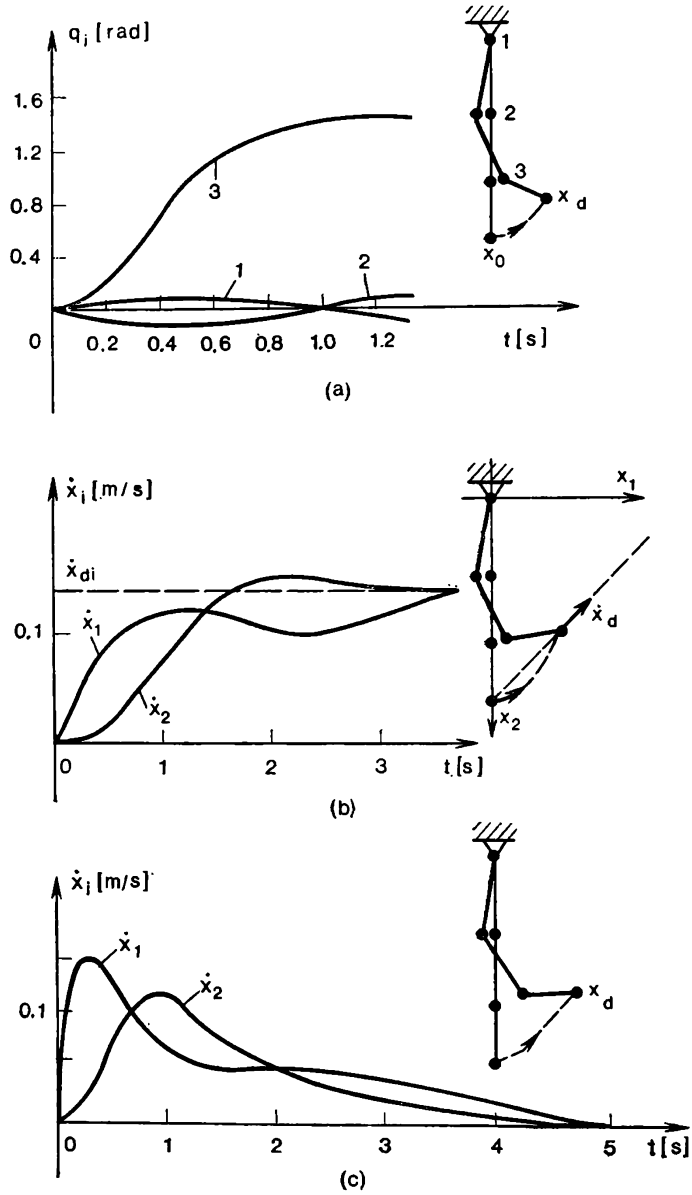


Fig. 8.6 Computer-simulated motion of a three-link manipulator in a vertical plane: a—position control; b—rate control; c—control providing zero speed at the assigned point

are the 6×1 vectors of the external and inertia forces and torques acting on the Nth link of the manipulator, and

$$\mathcal{F}_I' = [\mathbf{F}_{I1} \quad \mathbf{F}_{I\,N-1}0 \mid \mathbf{M}_{I1} \quad \mathbf{M}_{I\,N-1}0]$$

is a vector of inertia forces and torques of the robot links.

If the matrix $\mathbf{J}$ is square and nonsingular then to provide the required motion of the manipulated object with account of (8.32) it is enough to suppose

$$\mu = \mathbf{J}^{\mathrm{T}}(\mathcal{F} - \mathbf{R}), \tag{8.33}$$

where

$$\mathbf{R} = (\mathbf{J}^{\mathrm{T}})^{-1}\mathbf{B}^{0\mathrm{T}}\mathcal{F}_I'.$$

As the direct computation of the component $\mathbf{J}^{\mathrm{T}}\mathbf{R}$ is extremely machine time consuming, it can be determined approximately using the information obtained from the sensor of forces and torques applied to the hand

$$\mathcal{F}_N^* \approx \mathcal{F}_e + \mathcal{F}_{IN} \tag{8.34}$$

and also from the sensors of forces and torques actually developed in the joints

$$\mu^* = -\mathbf{J}^{\mathrm{T}}(\mathcal{F}_N^* + \mathbf{R}). \tag{8.35}$$

The equations (8.33)-(8.35) give

$$\mu = \mathbf{J}^{\mathrm{T}}\mathcal{F} + \mu^* + \mathbf{J}^{\mathrm{T}}\mathcal{F}_N^*. \tag{8.36}$$

Such a method of control is expedient when the external forces are insignificant in comparison with the manipulator inertia forces. Otherwise it is advisable to resort to a "dynamic regulation" control in the form

$$\mu = \mathbf{J}^{\mathrm{T}}(\mathbf{F} - \mathcal{F}_N^*). \tag{8.37}$$

8.5 ANALYSIS OF SEMIAUTOMATIC CONTROL SYSTEMS

The operation of semiautomatic control systems considerably depends on the ability of the human operator to receive information and to work out control signals. Nevertheless, considering the motion equations of the system operating in an automatic mode we can also obtain some recommendations for selecting control modes and corresponding parameters of control law.

We shall write the dynamic equations of the manipulator (8.23) in the form of differential equations expressing vectors $\mathbf{F}_I$, $\mathbf{M}_I$ in terms of derivatives of the generalized coordinates

$$\mathbf{A}(q)\ddot{q} = \mathfrak{B}(q, \dot{q})\dot{q} + \mathbf{B}_{(q)}^{0\mathrm{T}}[\mathbf{F}_e \mid \mathbf{M}_e]^{\mathrm{T}} + \mu. \tag{8.38}$$

The structure of the matrix $\mathbf{A}$ and partitioned matrix $\mathfrak{B}$ is described in [32]. The vector μ depends on the control law, control signals and also on the structure and parameters of actuators. Linearizing the motion equations we shall get conditions of the system stability with a chosen control law.

First we shall consider the problem of moving the hand to the target point using the algorithm of force control. Substituting the expression (8.31) with $\mathbf{u} = \mathbf{x}_d - \mathbf{x}$ in (8.38) we shall get

$$\mathbf{A}(\mathbf{q})\ddot{\mathbf{q}} = \mathfrak{B}(\mathbf{q}, \dot{\mathbf{q}})\mathbf{q} + \mathbf{B}^{0T}(\mathbf{q})[\mathbf{F}_e \mid \mathbf{M}_e]^T + \mathbf{J}^T(\mathbf{q})[(k_0(\mathbf{x}_d - \mathbf{x}) - k_1\mathbf{J}(\mathbf{q})\dot{\mathbf{q}}]. \tag{8.39}$$

Let us linearize this equation in the vicinity of the target point $\mathbf{x}_d = f(\mathbf{q}_0)$ with $\mathfrak{B}(\mathbf{q}, \dot{\mathbf{q}})$ linearly dependent on $\dot{\mathbf{q}}$ and $\mathfrak{B}(\mathbf{q}_0, 0) = 0$. Keeping in mind the relation $-(\mathbf{x}_d - \mathbf{x}) = \delta\mathbf{x} = \mathbf{J}(\mathbf{q})\delta\mathbf{q}$ and introducing the following notations $\mathbf{A}(\mathbf{q}_0) = \mathbf{A}_0$, $\mathbf{J}(\mathbf{q}_0) = \mathbf{J}_0$, $\mathbf{B}^0(\mathbf{q}_0) = \mathbf{B}_0^0$ we shall find

$$[\mathbf{A}_0 + \mathbf{A}_q(\mathbf{q}_0)\delta\mathbf{q} + o(\delta\mathbf{q})]\ddot{\delta\mathbf{q}} = [\mathbf{B}_q(\mathbf{q}_0, 0)\delta\mathbf{q} + \mathbf{B}_{\dot{q}}(\mathbf{q}_0, 0)\dot{\delta\mathbf{q}} + o(\delta\mathbf{q}, \dot{\delta\mathbf{q}})]\dot{\delta\mathbf{q}} + [\mathbf{B}_0^{0T} + \mathbf{B}_q^{0T}(\mathbf{q}_0)\delta\mathbf{q} + o(\delta\mathbf{q})][\mathbf{F}_e \mid \mathbf{M}_e]^T - \mathbf{J}_0^T k_0\mathbf{J}_0\delta\mathbf{q} - \mathbf{J}_0^T k_1\mathbf{J}_0\dot{\delta\mathbf{q}}. \tag{8.40}$$

Neglecting infinitesimals of higher order in the expression (8.40) we shall get

$$\mathbf{A}_0\ddot{\delta\mathbf{q}} = \mathbf{B}_0^{0T}[\mathbf{F}_e \mid \mathbf{M}_e]^T - \mathbf{J}_0^T\mathbf{J}_0[k_0\delta\mathbf{q} + k_1\dot{\delta\mathbf{q}}]. \tag{8.41}$$

Using former notations for variations of variables we shall have finally

$$\mathbf{A}_0\ddot{\mathbf{q}} = \mathbf{B}_0^{0T}[\mathbf{F}_e\ \mathbf{M}_e]^T - \mathbf{J}_0^T\mathbf{J}_0[k_0\mathbf{q} + k_1\dot{\mathbf{q}}]. \tag{8.42}$$

This is a system of differential equations of order 2N with constant coefficients. Let us investigate its stability.

To this end we shall write the equation of free motion as

$$\mathbf{A}_0\ddot{\mathbf{q}} = -\mathbf{J}_0^T\mathbf{J}_0[k_0\mathbf{q} + k_1\dot{\mathbf{q}}]. \tag{8.43}$$

Herein the matrix $\mathbf{A}_0$ is always positive definite, and the matrix $\mathbf{J}_0^T\mathbf{J}_0$ is symmetrical.

The matrix theory proves that there is a linear nonsingular transformation $\mathbf{P}$ such that $\mathbf{P}^T\mathbf{A}_0\mathbf{P}$ is an identity diagonal matrix and $\mathbf{P}^T\mathbf{J}_0^T\mathbf{J}_0\mathbf{P}$ is a diagonal matrix [4]. Substituting the variable $\mathbf{q} = \mathbf{P}\theta$ in the equation (8.43) we shall get

$$\ddot{\theta} + \lambda(k_0\theta + k_1\dot{\theta}) = 0, \tag{8.44}$$

where $\lambda = \text{diag}\,[\lambda_i]$ are the roots of the characteristic equation

$$\det\,[\mathbf{J}_0^{\mathrm{T}}\mathbf{J}_0 - \lambda\mathbf{A}_0] = 0. \tag{8.45}$$

We shall denote eigenvalues of the matrix $\mathbf{A}_0^{-1}\mathbf{J}_0^{\mathrm{T}}\mathbf{J}_0$ as λ_i.

The equation (8.45) shows that the trivial solution of the system (8.43) is stable if and only if all λ_i are strictly positive. From this equation we obtain solutions θ_k in the form

$$\theta_k = a_k \exp(-\gamma_k t)\sin(\omega_k t + \alpha_k), \tag{8.46}$$

where

$$\gamma_k = k_1\lambda_k/2,\quad \omega_k = (\lambda_k k_0 - (k_1\lambda_k)^2/4)^{1/2}.$$

Herein the parameters a_k, α_k are determined by the initial conditions θ_{k0}, $\dot{\theta}_{k0}$.

The expression (8.46) shows that at $k_1 = 0$ the solution has the form of periodic oscillations. The original system $\mathbf{q} = \mathbf{P}\theta$ forms a sum of periodic oscillations with nonmultiple frequencies, i.e. nonperiodic oscillations. Such a motion corresponds to a strict application of force control, i.e. to a selection of the control algorithm by the formula (8.30) when $k_1 = 0$. To provide an asymptotical stability of the control system the component $-k_1\dot{\mathbf{x}}$ damping the manipulator motion is added to the formula (8.30).

From the expression (8.46) a conclusion can be drawn that with

$$k_1 \geqslant 2\,(k_0/\min_k \lambda_k)^{1/2} \tag{8.47}$$

all the solutions θ_k have an aperiodic character and with $k_1 < 2\,(k_0/\min_k \lambda_k)^{1/2}$ they have the form of damped oscillations.

Therefore the solution $\mathbf{q}$ of the system of differential equations (8.43) will be aperiodic, if the inequality (8.47) is satisfied, and oscillatory otherwise.

An increase in the system's inertia determined by the coefficients of the matrix $\mathbf{A}_0$ results in the oscillation frequency decrease; while an increase of the coefficients of the matrix $\mathbf{J}_0$ leads on the contrary to oscillation frequency increase. This effect can be physically interpreted as an increase of the mechanical system rigidity.

In some points of the manipulator operating space the matrices $\mathbf{J}_0$ and $\mathbf{A}_0^{-1}\mathbf{J}_0^{\mathrm{T}}\mathbf{J}_0$ may become singular, i.e. there is at least one among the eigenvalues λ_i equal to zero. The equation (8.44) shows that the trivial solution of the corresponding differential equation will be unstable and with $\dot{\theta}_k(0) \neq 0$ we shall get $\theta_k(\infty) \to \infty$. Hence, the solution of the original system of differential equations is also unstable in the vicinity of the specified point corresponding to the singular matrix $\mathbf{J}_0$.

From this analysis the following practical conclusions can be inferred.

First, as the eigenvalues of the matrix $\mathbf{A}_0^{-1}\mathbf{J}_0^{\mathrm{T}}\mathbf{J}_0$ are real and nonnegative, the motion of the system will be stable throughout the entire operating space with the exception of singular points where $\det\mathbf{J}_0 = 0$. They can be neglected, however, if the probability that the robot should be maintained at these points is small. The system will not lose stability therein because the algorithm of force control does not require conversion of the matrix $\mathbf{J}$. Otherwise, the control algorithm is to be selected from the formula

$$\boldsymbol{\mu} = \tilde{\mathbf{J}}^{\mathrm{T}}(\mathbf{q})\,[k_0\,(\mathbf{x}_{\mathrm{d}} - \mathbf{x}) - k_1\tilde{\mathbf{J}}(\mathbf{q})\,\dot{\mathbf{q}}], \tag{8.48}$$

where $\tilde{\mathbf{J}}(\mathbf{q})$ is the matrix coinciding with $\mathbf{J}(\mathbf{q})$ in regular points and having the inversed one in singular points of $\mathbf{J}(\mathbf{q})$. If $\mathbf{B} = \tilde{\mathbf{J}}^{-1}(\mathbf{q})$ is the matrix obtained as a result of an approximate inversion of the matrix $\mathbf{J}(\mathbf{q})$ (see Sec. 2), then $\tilde{\mathbf{J}}(\mathbf{q}) = \mathbf{B}^{-1}$.

Second, the condition (8.37) can be used to approximately assign the parameters k_0, k_1 of the control law, corresponding to the transient. Using the known relations for transient time T_{t} and oscillations μ we shall have

$$T_{\mathrm{t}} \leqslant \frac{1}{\gamma}\ln\frac{1}{\Delta}, \tag{8.49}$$

where $\gamma = \min k_1\lambda_i/2 = k_1\lambda_{\min}/2$, and Δ is the parameter usually taken from 0.01 to 0.05;

$$\text{and } \mu \geqslant \max_k \omega_k/\gamma_k = \sqrt{\frac{4k_0}{k_1^2\lambda_{\min}} - 1}. \tag{8.50}$$

From the first inequality we shall find

$$k_1 \leqslant \frac{2}{T_{\mathrm{t}}\lambda_{\min}}\ln\frac{1}{\Delta}, \tag{8.51}$$

The second inequality gives

$$\frac{k_1}{k_0} \geqslant \frac{4}{\lambda_{\min}(1+\mu^2)}.$$

With account of (8.49) we shall have

$$\frac{k_1}{k_0} \geqslant \frac{2T_{\mathrm{t}}}{(1+\mu^2)\ln 1/\Delta}. \tag{8.52}$$

We can select from the inequality (8.51) k_1 taking into account an assigned time of the transient T_{t} and $\lambda_{\min}$, and (8.52) gives k_0. We would like to note that the evaluation of the ratio k_1/k_0 is independent of the eigenvalues λ_i and is defined by the assigned requirements to the transient.

Now we shall discuss the algorithm of rate control while moving the hand to the point $\mathbf{x}_d$. To describe the dynamics of the actuators we shall use a simplified equation

$$\mu_k = k_0 [q_{k0}(t) - q_k(t)] - k_1 \dot{q}_k(t), \tag{8.53}$$

where $q_{k0}(t)$ is the control signal.

We shall take into account that in the vicinity of the target point

$$\delta \mathbf{q}_0 = k\tilde{\mathbf{J}}^{-1}\delta \mathbf{x} \approx k\delta \mathbf{q}. \tag{8.54}$$

Substituting expressions (8.43) and (8.44) into the equation of the actuator and linearizing it as has been shown above we get

$$\mathbf{A}_0 \ddot{\mathbf{q}} = \mathbf{B}_0^{0T} [\mathbf{F}_e \vdots \mathbf{M}_e]^T - [k_0 k\mathbf{q} + k_1 \dot{\mathbf{q}}]. \tag{8.55}$$

Wherefrom the equation of free motion will assume the form

$$\mathbf{A}_0 \ddot{\mathbf{q}} = -[k_0 k\mathbf{q} + k_1 \dot{\mathbf{q}}]. \tag{8.56}$$

Let $\mathbf{P}$ be a linear nonsingular transformation which diagonalizes the matrix $\mathbf{A}^{-1}$, then substituting $\mathbf{q} = \mathbf{P}\theta$ in the last equation we shall have

$$\ddot{\theta} + \lambda [k_0 k\theta + k_1 \dot{\theta}] = 0. \tag{8.57}$$

Here λ is a matrix of roots of the characteristic equation $\det [\mathbf{E} - \lambda \mathbf{A}_0] = 0$, i.e. of the eigenvalues of the matrix $\mathbf{A}_0^{-1}$.

The equation (8.56) is a particular case of the equation (8.43) when the matrix $\mathbf{J}_0$ is orthogonal. Therefore all the reasoning pertaining to the equation (8.43) is also valid for (8.56).

The equilibrium is stable when λ_i are strictly positive and the solutions θ_i have the form of (8.46). If $k_1 \geqslant 2 (k_0/k\lambda_{\min})^{1/2}$ the solutions are aperiodic; while with $k_1 \approx 0$ continuous oscillations occur in the system.

We shall note that the equation (8.57) is not satisfied in the vicinity of singular points where $\det \mathbf{J}_0 = 0$. Really the approximate equality (8.54) in this region is specified as follows:

$$\delta \mathbf{q}_0 \approx -k\tilde{\mathbf{J}}^{-1}\mathbf{J}\delta \mathbf{q}$$

with

$$\tilde{\mathbf{J}}^{-1}\mathbf{J} \neq \mathbf{E}, \quad \det \tilde{\mathbf{J}}_0 \neq 0.$$

Then instead of the equation (8.56) we shall have

$$\mathbf{A}_0 \ddot{\mathbf{q}} + k_0 \tilde{\mathbf{J}}_0^{-1}\mathbf{J}_0 \mathbf{q} + k_1 \dot{\mathbf{q}} = 0. \tag{8.58}$$

This equation determines the robot motion in the vicinity of a singular point. As in the previous case, the analysis shows that the system is not stable in this region.

While comparing algorithms of force and rate control it may be inferred that the accepted terminology is rather conditional because the rate control algorithm enables the robot to develop certain force and that of force control permits the robot to realize motion at an assigned speed. Rate control, however, demonstrates the existence of a considerable error between the vector of control handle deflection and the force vector developed by hand. In a static mode of operation as the expressions (8.53) and (8.54) show, the moments of position-feedback actuators are proportional to control signals, i.e.

$\boldsymbol{\mu} = k\tilde{\mathbf{J}}^{-1}(\mathbf{q})\,\mathbf{u};\ k = \text{const},$

then

$$\mathcal{F} = k\,(\mathbf{J}^{\mathrm{T}}(\mathbf{q})\,\tilde{\mathbf{J}}^{\mathrm{T}}(\mathbf{q}))^{-1}\mathbf{u}. \tag{8.59}$$

The operator $(\mathbf{J}^{\mathrm{T}}\tilde{\mathbf{J}})^{-1}$ determines the error.

For force control of motion at a constant speed, the following relation can approximately be written

$$\begin{aligned} \dot{\mathbf{q}} &= \mathrm{k}'\boldsymbol{\mu} = k'\mathbf{J}^{\mathrm{T}}\mathbf{u};\ k' = \text{const}, \\ \dot{\mathbf{x}} &= k'\,(\mathbf{J}\cdot\mathbf{J}^{\mathrm{T}})\,\mathbf{u}. \end{aligned} \tag{8.60}$$

Thus, the operator $\mathbf{J}\mathbf{J}^{\mathrm{T}}$ characterizes an error between the hand velocity vector and the control handle deflection. From the above reasoning it follows that to control robot manipulations requiring high accuracy of motion along the path, i.e. in a zone of numerous obstacles, in welding operations, etc., it is expedient to use the algorithms of rate and position control (the latter given in Section 2).

While performing jobs which require control of forces and torques, i.e. operations with kinematically constrained objects, control should be implemented in a force mode.

As follows from relations (8.53) and (8.60), the control vector $\dot{\mathbf{x}}$ or $\mathcal{F}$ is proportional to the control handle deflection vector $\mathbf{u}$ irrespective of the chosen control algorithm when and only when the matrix $\mathbf{J}$ is orthogonal. The orthogonality conditions, in particular, can be satisfied for minors of the matrix $\mathbf{J}$ in some domains of the space of the generalized coordinates $\mathbf{Q} \supset \mathbf{q}$. Then there are domains in the space of vectors $\mathbf{U} \supset \mathbf{u}$ for which the control vector $\dot{\mathbf{x}}$ or $\mathcal{F}$ is proportional to the vector $\mathbf{u}$ [56]. In these domains one and the same control algorithm can be successfully used for solving various manipulation problems. For example, for a robot controlled in a rate mode, assembly operations should be planned in that work-space where the orthogonality conditions are satisfied. A more general approach is to use the properties of a symmetric matrix $\mathbf{J}\mathbf{J}^{\mathrm{T}}$. In each

point **q** where it is nonsingular there exists an orthogonal set of eigenvectors $\mathbf{d}_i$ corresponding to real terms λ_i, $i = 1, \ldots, 6$. If the vector **u** coincides with $\mathbf{d}_i$ then

$$\begin{aligned} \dot{\mathbf{x}} &= k'\lambda_i \mathbf{u}, \\ \mathcal{F} &= k/\lambda_i \mathbf{u}, \end{aligned} \tag{8.61}$$

and the direction of the control vectors **x**, $\mathcal{F}$ coincides with that of the vector **u**. Using redundant robot joints, or moving its base, the vector **u** can be aligned with one of the eigenvectors $\mathbf{d}_i$, which insures proportionality of the vector **u** and control vector. Such a technique can considerably increase the accuracy and efficiency of manipulations in a semiautomatic system.

The dynamics of a semiautomatic system with the human operator can be tested only experimentally. For this not only real-life robots but also their mathematic models can be used. In both cases the human operator must work under conditions similar to those of normal operation using real control devices and information displaying means.

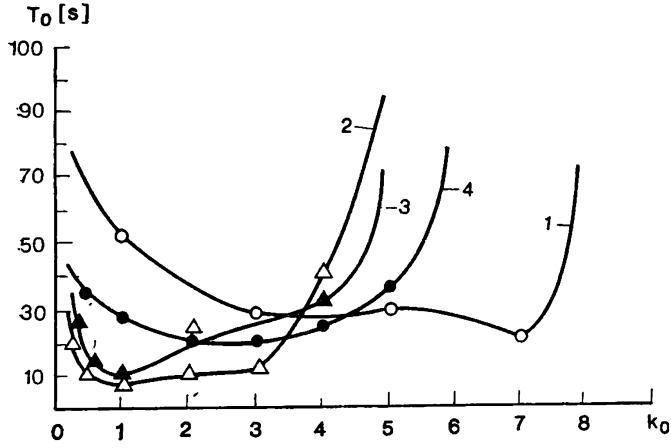


Fig. 8.7 Simulation for different control laws of a behavior of a semiautomatic system controlled by the human operator

Simulating the behavior of a semiautomatic system controlled by the human operator gives more precise information on how to select correctly parameters of a control law. Figure 8.7 shows the results of such a simulation for different control laws. The robot parameters were taken as in the example given in Section 4. The abscissa is the scale coefficient k_0 entering the formulas (8.14) and (8.31), and the ordinate is the time of moving the hand into a specified point. Curves *1* and *2* correspond to automatic control, the first one to force

control and the second to rate control. The first technique is more time consuming, however, but ensures serviceability over a wider range of the coefficient k_0. Curves *3* and *4* correspond to semiautomatic control. Participation of the human operator in force control (curve *4*) increases the speed of system's response but reduces the operating range of the coefficient k_0 (as compared with curve *1*). In rate control (curve 3) the system's operation resembles that in automatic control because the nature of relationships is preserved in a greater robot workspace and they are distorted only in the vicinity of singular points.

The performance of semiautomatic systems is essentially enhanced if forces and moments acting on the manipulator hand are displayed on a control device. The human operator senses the forces unknown in advance and then selects a control vector. A force-feedback is especially essential for operations with kinematically con-

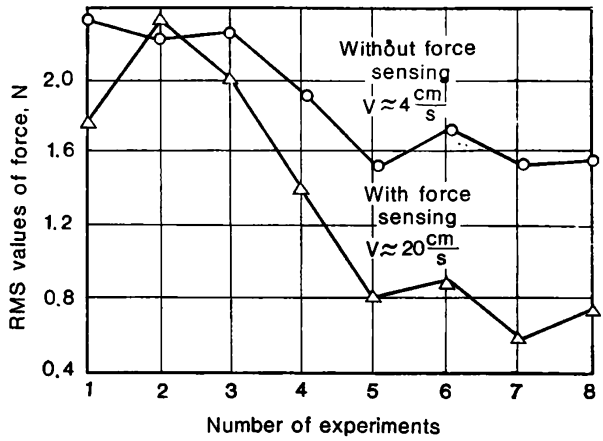


Fig. 8.8 Curves of force sensing in force controlled system

strained objects. Figure 8.8 shows an effect of sensing force when force-controlled robot turns a lever. With force-feedback an RMS value of the reaction forces is reduced by more than twice. The operation rate increases fivefold.

While selecting requirements for precision of sensors of forces and torques, transducers and computers one should bear in mind that the human operator's receptors possess a considerably small precision of sensing force. Figure 8.9 shows error probability graphs and operator's reaction time in sensing force. In the range of 8-20 N the operator can distinguish not more than 5 grades of force magnitude.

The perception threshold in this range is not less than 3 N. Figure 8.10 represents error probability graphs in recognizing the force

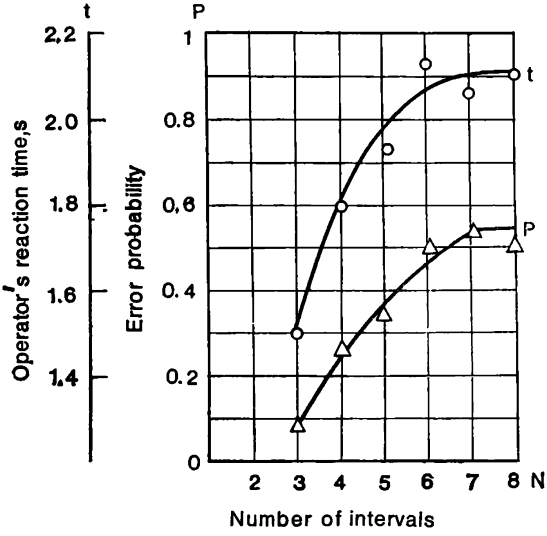


Fig. 8.9 Graphs of error probability and operator's reaction

vector direction and the operator's reaction time. The human operator reliably distinguishes not more than 12 directions, i.e. the pre-

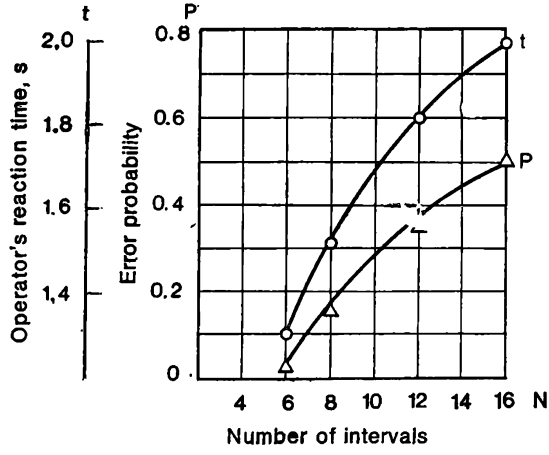


Fig. 8.10 Graphs of an error probability in recognizing force vector direction, and operator's reaction time

cision of his operation is approximately 30°. This allows us to reduce the requirements to the precision of the devices sensing and reproducing power information. Nevertheless semiautomatic control in-

sures a sufficiently high performance of the system due to the adaptive properties of the human operator.

With considerable and nonpredictable external forces the work of the human operator may be facilitated by providing force-feedback in the control system proper, i.e. by selecting the vector μ according to the formula (8.37)

$$\mu = \mathbf{J}^{\mathrm{T}} (\mathcal{F} - \mathcal{F}_N^*), \tag{8.62}$$

where $\mathcal{F}_N^*$ is the vector of forces and torques measured by a sensor in the wrist. The efficiency of this method is illustrated in Fig. 8.11 on

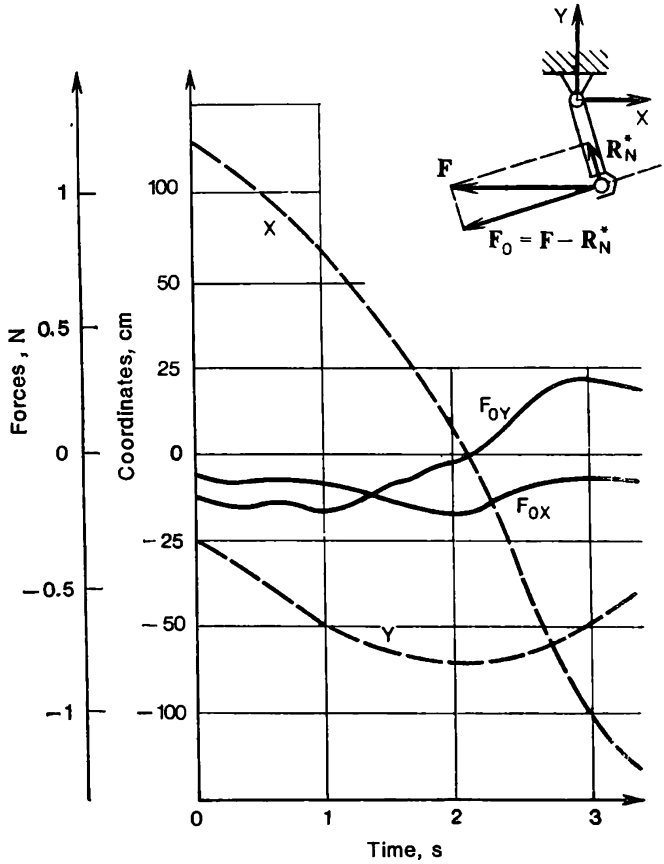


Fig. 8.11 Human operator's efficiency in a robot control system with power feedback

the example of a lever turn operation. Throughout the operation the control effect $\mathcal{F}$ applied by the operator has the same direction. Nevertheless force-feedback makes the actual effect applied to the

lever by the robot, be directed close to the tangent line, which insures quick and smooth fulfilment of the operation.

These preliminary results of simulation prove the assumptions which were taken as the basis of selecting control algorithms in the preceding Sections. At the same time the human operator essentially modifies the system operation, and this should be taken into consideration while designing semiautomatic systems. The preliminary results of the research allow us to plan the ways of solving a number of problems of modern robotics.

List of Symbols

$\mathbf{q}$	vector of generalized manipulator coordinates
$\mathbf{x}$	vector of manipulator hand position and orientation
$\mathbf{J}$	Jacobian matrix
$\mathbf{e}$	unit vector of a revolute kinematic pair
$\mathbf{h}$	unit vector of a prismatic kinematic pair
$\mathbf{R}$	position vector
$\mathbf{I}$	quality functional
$\dot{\varphi}$	angular speed of the hand absolute speed vector
$\dot{\psi}$	angular speed of sighting
τ	orthogonal transition matrix
$\mathbf{V}$	hand velocity vector
$\mathbf{D}$	distance vector
$\mathbf{F}_e$, $\mathbf{M}_e$	vectors of external forces and moments acting on the manipulator links
$\mathbf{F}_I$, $\mathbf{M}_I$	vectors of the manipulator main inertia forces and moments
μ	vector of forces and moments developed by the actuators

9 Synthesis of Walking Motion Control

D. E. Okhotsimsky, *Corr. Mem. USSR Acad. Sc.*,
A. K. Platonov, *Cand. Sc. (Phys.-Math.)*,
E. A. Devyanin, *D. Sc. (Phys.-Math.)*, and
V. S. Gurfinkel, *D. Sc. (Biol.)*

9.1 GENERAL

The accumulated experience of elaborating methods of walking motion control shows that modern computer facilities are capable of controlling statically stable motion of a six-legged man-supervised walking vehicles. The use of hybrid digital/analog control system holds much promise in controlling motion of walking vehicles. The paper describes models of walking machines and their control systems.

9.2 WALKING VEHICLE AS A COMPLICATED ROBOTIC SYSTEM

The motion activity of a walking vehicle is sufficiently complicated and multifunctional. It comprises motions in space, motions insuring active perception of the environment, motions providing stability and safety of the machine while executing various working operations. In general the walker moves under the conditions indefinite with respect to the environment, its geometric and bearing characteristics.

A developed walker must have all the properties of the robotic system combining functional elements sensing the state of its parts and environment, making decisions on the required actions, planning and executing motions. A walking robot possesses certain autonomy in organizing joint operation of subsystems and parts which allows the human operator to simplify considerably its control.

A complicated motion of a walker results in a hierarchy structure of planning a control system. The lowest level represents a servo motion program. Then follow plans of changing the support and swing phases within a stepping cycle for predetermined feet positions. Furtheron follows a plan of leg motion coordination (or coordination of their stepping cycles) determining gait of the vehicle. Next hierarchy level defines slants and turns of the walker body and also choice of the motion direction and speed.

The operator can interfere with any level of motion planning. The plans shaped on the higher level determine the planning process on the lower levels. The upper hierarchy levels should be provi-

ded with resources and information for a longer period of time than the lower ones.

Development of a high-performance walking robot pertains to designing a complex system. One of the objectives strived at is to create an organization reducing the information flow in the system. This goal is closely connected with rationally correcting the robot motion plans.

Robot motion requires information feedback channels from the environment. They must correct the robot motions in reference to task problems.

Motion correction primarily necessitates evaluating the environment geometry. The most convenient herein is the measurement of distances from the robot to various points in his world. This can be performed by direct location, or by means of triangulation or stereoscopic methods of distance measuring.

Processing the information received from the environment can be interpreted as a pattern recognition problem. The objective is to construct a generalized image representing only those features of the environment which are essential for motion.

Control and correction of robot motion is to a certain extent a problem inverse to that of processing the measurements. Motion should be constructed using a sufficiently general representation of the environment.

Motion control of mechanical systems with a great number of degrees of freedom is not an ordinary problem of applied engineering. In this connection mention should be made of the study carried out by N. A. Bernstein [1]. While analyzing various phases of man motion he formulated a remarkably simple postulate, which states that the main objective of motion coordination is to overcome a redundancy of the degrees of freedom in the system. With reference to walking vehicles this concept means investigating the ways of reducing the number of independent variables controlling motion.

Control of the walkers locomotion is mainly aimed at solving a problem of statically stable motion of a six-legged walking vehicle [35]. But even motion within this mode raises the problem of the walking robot dynamics. The dynamic aspects of motion must be primarily considered while developing a leg servo control system and also planning motion at a small stability margin.

To analyze the six-legged robot dynamics is a sufficiently complicated problem due to a great number of degrees of freedom in the system. There are 24 degrees of freedom altogether, 18 of them being in the leg joints and 6 in the robot body. The motion control algorithms are evaluated using a vehicle dynamic diagram taking account of the mass-inertial characteristics of the body and legs. In a number of cases a simplifying assumption is permissible, i.e. to

neglect the leg inertia which is small as compared to that of the servos.

A six-legged walking vehicle is a statically indefinite system since six equations of statics must balance from 9 to 18 components of support reactions. This necessitates wise equilibration of the forces. To get out of a difficulty, included into the control system are sensors of forces and torques measuring the support reactions. They are to be distributed over legs of the walker depending on the locomotion mode and terrain properties so as to ensure the greatest leg/terrain cohesion or the smallest specific load on the leg carrying elements. Wise distribution of support reactions reduces a senseless "counteraction" in the leg servos and thus decreases energy consumption.

Walking vehicles were designed by computer simulation methods and by way of physical modeling.

9.3 SIMULATION OF CONTROL ALGORITHMS

Simulation of control algorithms is realized in program complexes written in AUTOCODE, MACRO, REFAL, ASTRA and SIMULA.

Working out the algorithms comprises a simulation of the walker itself which is the object of control and the environment with which it interacts during the motion. Also simulated is a process of collecting data from the robot's sensors and metering systems and its subsequent conversion and processing in a control system. To this end we must devise a mathematical model of the control system computation means and the structure of data conversion and flow between the control system processors. And finally, simulation of the energy requirements of the walking vehicle helps optimize the structural characteristics and motion modes of the walker.

The amount of detail accumulated in research essentially complicates the simulation studies and starting from some level of detailization it seems advisable to use laboratory models to investigate the walking vehicles design. The greatest difficulty, however, presents mathematical modeling of the environment, which in the experimental research can be omitted.

At this stage of mathematical modeling of vital importance was graphic representation of a walker and terrain on the display screen since it allowed one to evaluate the algorithm performance. To this end a problem-oriented software has been devised.

Earlier simulation studies described in [8, 9, 12, 35, 36, 38, 61] were devoted to kinematic considerations, i.e. to analyzing gaits and elaborating motion algorithms. The results of the research are presented below.

Kinematic capabilities of a walking vehicle with dimensions as in Fig. 9.1 are determined by the maximum size of the overcome obstacles (tabulated below) devised by computer simulation.

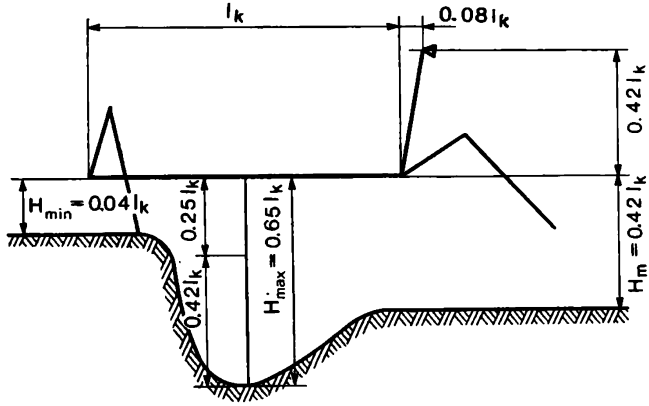


Fig. 9.1 Dimensions of a walker

Obstacle	Width	Depth	Height	Notes
Crevasse	1.04 lk	> H max		Equihigh
Crevasse	0.25 lk	> H max	1.3 H max	Equihigh
Step+			1.3 H max	Upward
Step—		1.3 H max		Downward
Stone	0.75 lk		0.71 H max	Stepping upon is prohibited
Stone	> 0.25 lk	1.3 H max	0.71 H max	Stepping upon is allowed

While considering six-legged gaits, mention should be made of the entomological research [63] which provided a great body of information on a wave gait of insects. The application of results to the analysis of wave six-legged gaits of walking vehicles [37] allowed the researchers to obtain a relation between the speed and static stability margin for all regular and symmetric gaits (Fig. 9.2). However, simulating the algorithms of coordination of leg motion over uneven relief presenting some limitations for leg locomotion made the researchers abandon the biological prototypes and synthesize a free gait algorithm [58] providing the best cross-country capacity and stability of a walker.

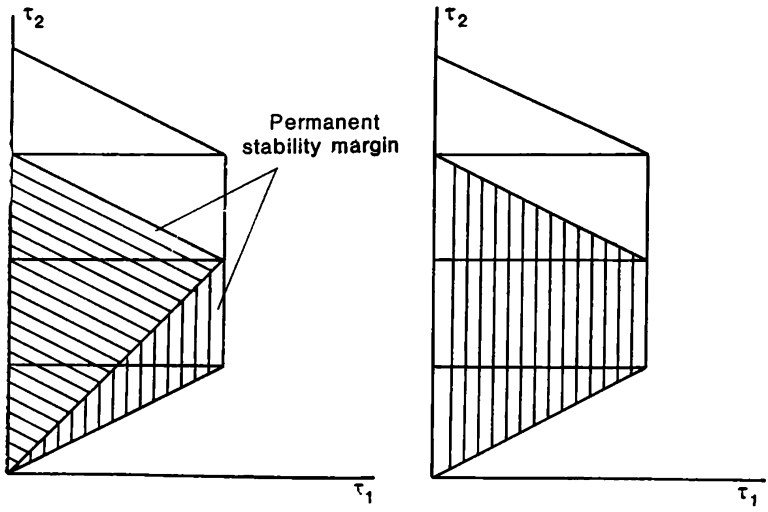


Fig. 9.2 Regular symmetric wave gaits

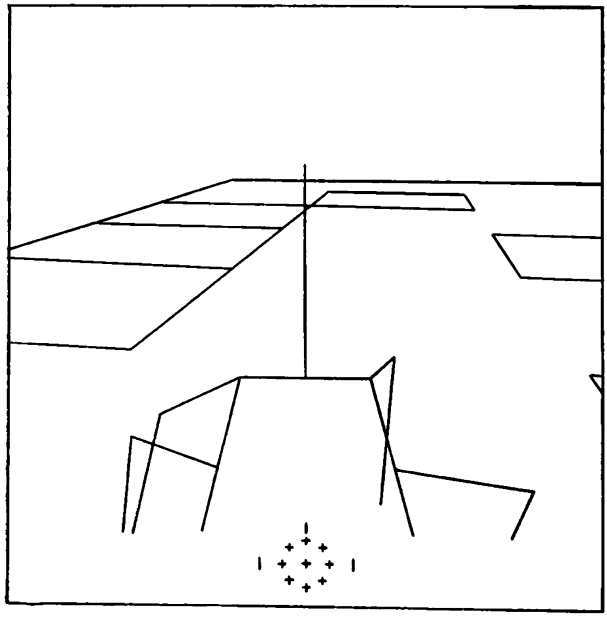


Fig. 9.3 Environment image on a display screen

Motion of a six-legged walker is constructed [1] as a two-level system of algorithm modules. The upper level is represented by a monitor of the motion construction system and the lower one comprises a set of algorithms of elementary problems of motion construction.

The monitor determines a sequence of elementary algorithms depending on the results of program functioning and environment data.

For the human operator to accomplish his functions control algorithms were devised [37] simulating control system operation in real time. Feedback is furnished by using a light pen and two-degrees-of-freedom joystick. The state of terrain is represented on the display screen (Fig. 9.3). A large volume of incoming information cannot be percepted by the human operator, which necessitates automation of legged locomotion, i.e. selection of footprints and locomotion speed and direction. The main difficulty in organizing a supervisory control lies in that the control system cannot prognose the driver's actions. This imposes limitations on vehicle manoeuvrability depending on the motion phase, and motion is constructed so as to prohibit driver's actions violating the limitations. To lift these an algorithm of driver's actions is created.

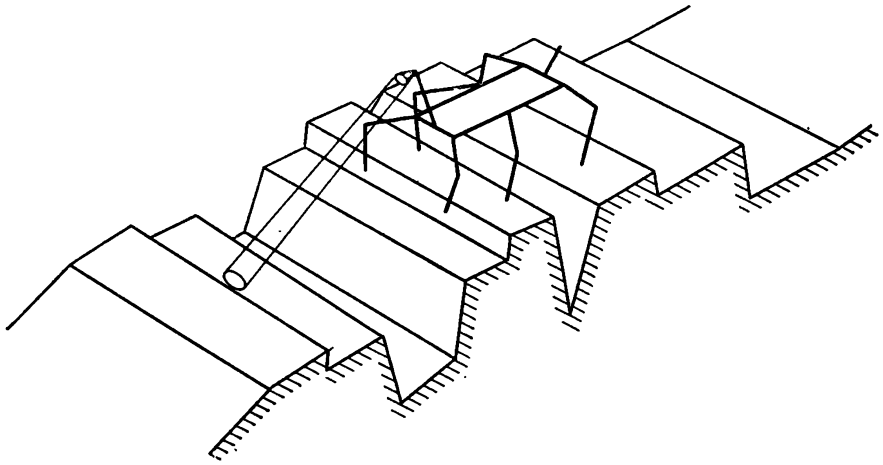


Fig. 9.4 Motion over prismatic terrain

The algorithms of controlling an integral robot with range-metering system [6] were first developed with reference to moving along a straight path covered with prismatic obstacles (Fig. 9.4). Described below are the principles of interaction of the system of motion construction and that of data gathering elucidated in the research. Active surveying of terrain through manoeuvering a metering beam

enables the vehicle to automatically overcome sufficiently complicated obstacles with parameters close to kinematically possible ones. Figure 9.5 shows a display image of a vehicle which having approached a deep crater surveys it and then overcomes it, the metering

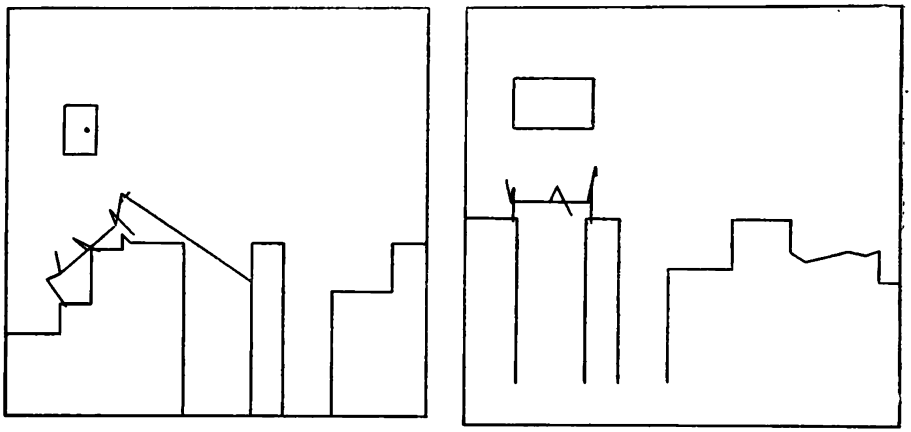


Fig. 9.5 Simulating algorithms for overcoming difficult terrain: a—slant motion; b—horizontal motion

system being switched off until arriving at the next position for surveying. A rectangle with a dot inside given in the upper part of the figure shows the position of the walker's center of mass inside the support polygon and is used to evaluate a static stability margin in the locomotion.

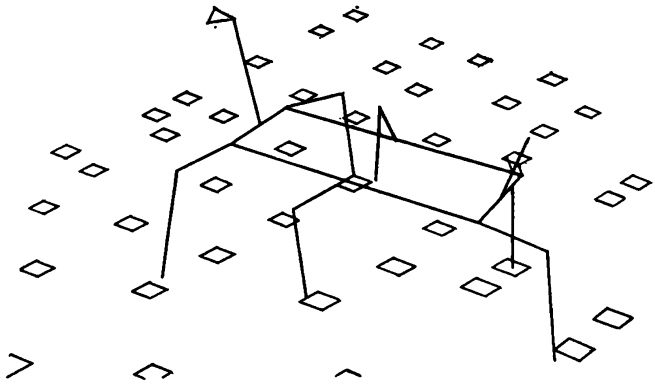
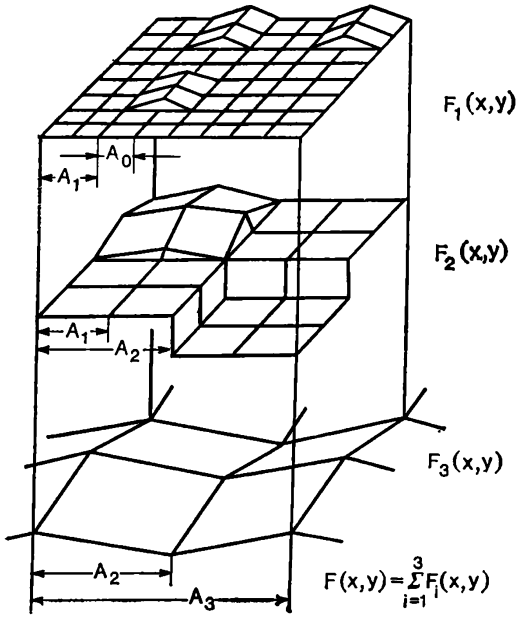


Fig. 9.6 Motion over terrain with limited surface for feet placing

Motion simulation in a three-dimensional space presents more difficulties. Motion in a medium providing small capabilities for foot placing is simulated, as shown in Fig. 9.6, by means of graphic

displaying. The devised algorithms ensure a walking vehicle motion with maximum stability margin and automatic selection of support and swing phases and points for feet placing on the ground. The



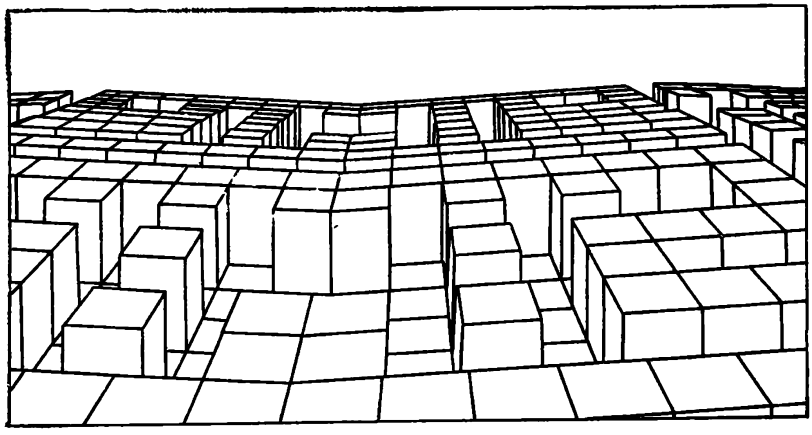


Fig 9.7 Model of terrain with obstacles

support platforms are irregularly located and are not less than 4% of the total area.

To synthesize algorithms of locomotion over uneven relief we developed a model of terrain [41] based on the use of three levels of a

square grid inserted one into another, each carrying some standard objects assigned thereupon. The model, on the one hand, represented difficult terrain and, on the other, required relatively small computer resources on its storing, access and displaying with erasing invisible lines (Fig. 9.7).

We also attached great priority to the development of algorithm complexes providing locomotion of a legged vehicle mathematical

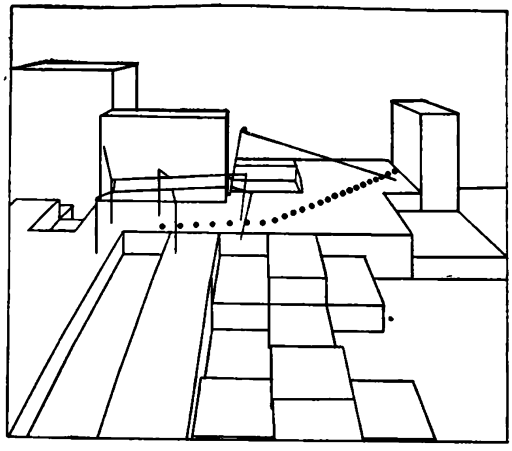


Fig. 9.8 Simulating algorithms for selecting a motion path

model which automatically gathers information on terrain along an assigned path and measures it with the purpose to avoid large obstacles (Fig. 9.8).

Mathematical modeling of a walking machine produces logic and algorithms of those levels of the legged vehicle control system which do not control data transducers and leg servos. The latter were developed by physical modeling.

9.4 PHYSICAL MODELING OF WALKING VEHICLES

In physical modeling the control algorithms are realized by means of software/hardware complex operating in a real time mode, which requires strict elaboration of control algorithms, and interfaced to laboratory models of walkers, data transducers and obstacles of the environment. Discussed below in brief are some stages of physical modeling and construction of laboratory models of walking vehicles.

From the very beginning the researchers tackled possible ways of developing control equipment both on the basis of digital and analog computers. The first stage of physical modeling involved the use of universal digital and analog computers (Fig. 9.9), which allowed the composition and structure of the algorithmic part of control system

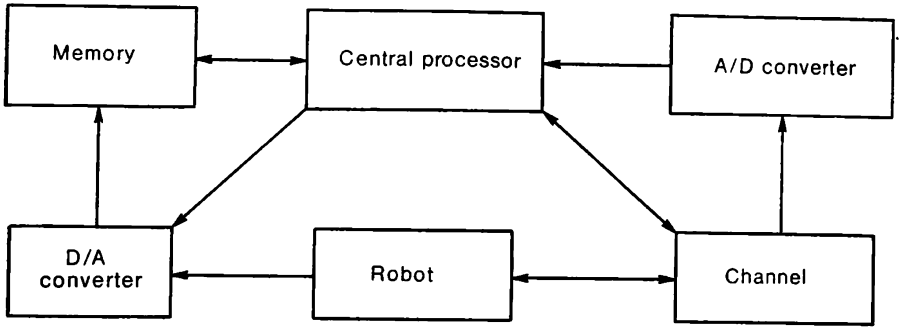


Fig. 9.9 Simulation hardware

to be altered in a comparatively simple way. The next stage was a development of the control system models of walking vehicles in the form of dedicated digital and analog hardware. This resulted in

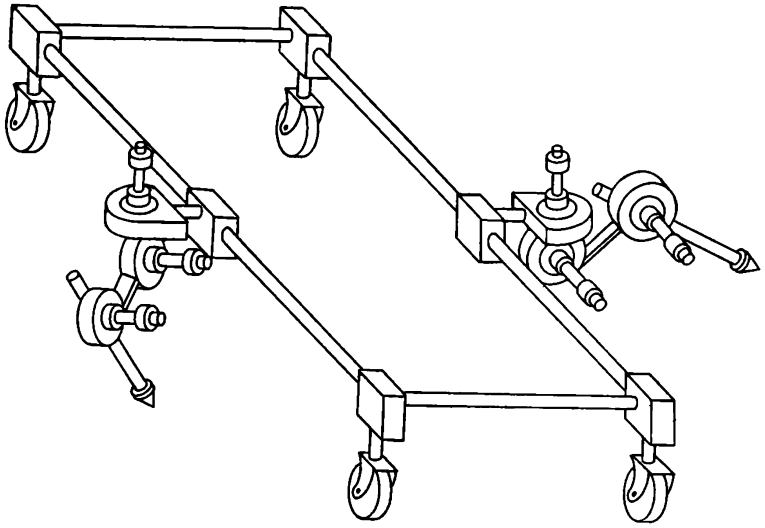


Fig. 9.10 Model for studying interlimb coordination

elaboration of corresponding mass-size characteristics of the legged locomotion control system.

The first models (see Fig. 9.10) were designed to devise means and methods of controlling leg motion. The machine could move along a

rectilinear and curvilinear path at different speeds assigned by the operator. It was used to test coordination of leg motion adopted from physiology and hardware employing off-the-shelf elements of a quasi-neuron type.

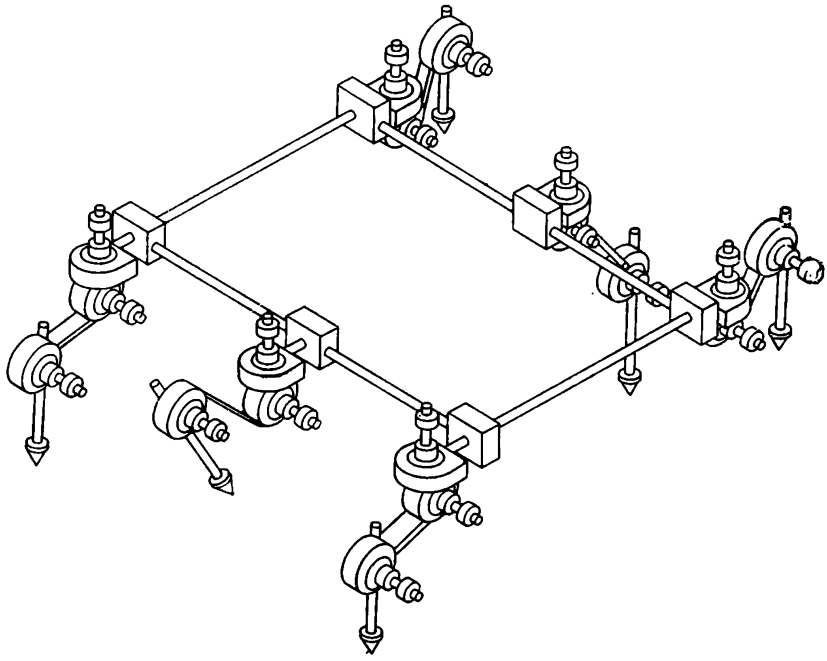


Fig. 9.11 Model for developing walking vehicle gaits

Figure 9.11 [8] portrays the first attempt to realize the actuator system of a six-legged vehicle. This model can simulate regular wave gaits of several types and helps formulate requirements to the load capacity of the legged machine.

The analog computation means was used to work out the logic of leg motion control by means of three sequentially modified "ricksha" type machines (Figs. 9.12-9.14) [6, 35, 41]. At the initial stage of research the machine configuration allowed us to disregard locomotion stability while synthesizing algorithms of controling the leg servos in overcoming relatively large obstacles. The "ricksha" model shown in Fig. 9.13 was used to develop the elements of telereception comprising a double-linked probe with an annular contact sensor and potentiometers in the joints. As experience in modeling and mathematical simulation has shown, this simple mechanical device can evaluate the size and shape of obstacles, plan footprints and phases of obstacle clearing.

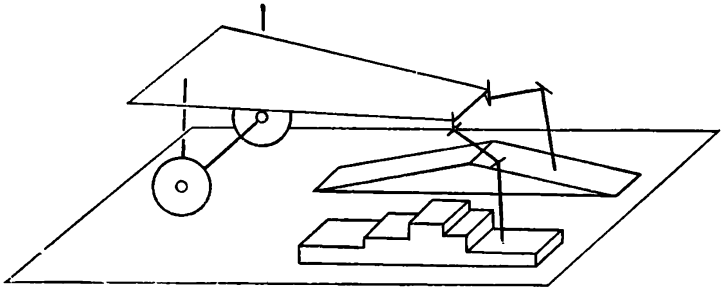


Fig. 9.12 Ricksha-type model for developing leg actuators

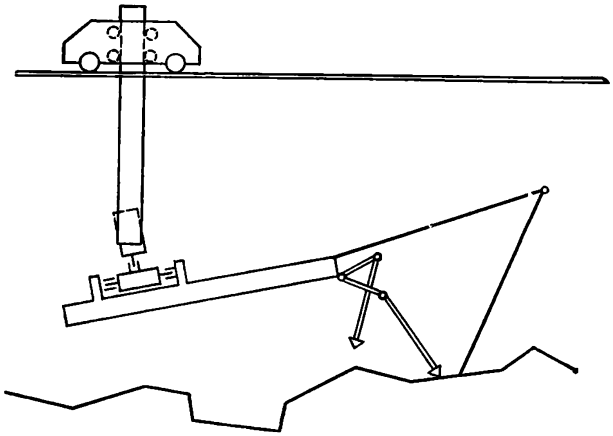


Fig. 9.13 Ricksha-type model for developing motion over uneven terrain

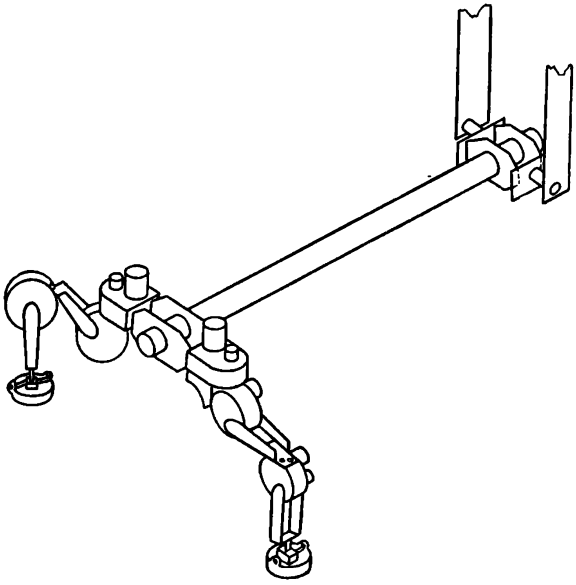


Fig. 9.14 Ricksha-type model for developing leg-stroke generators

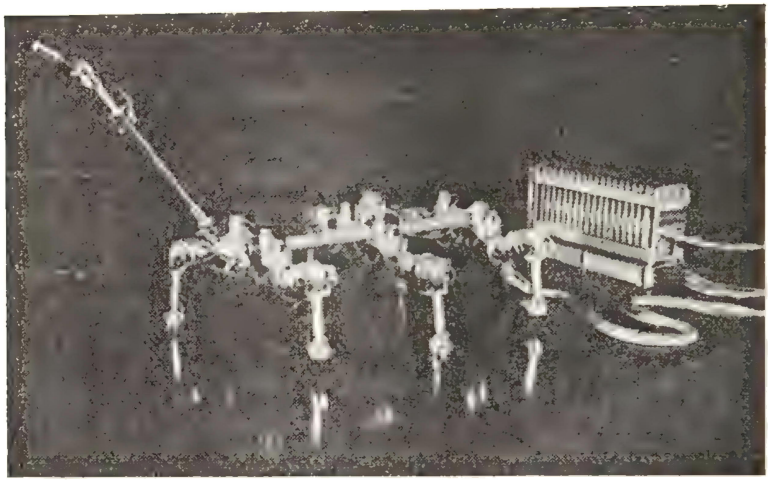


Fig. 9.15 Six-legged analog controlled walker

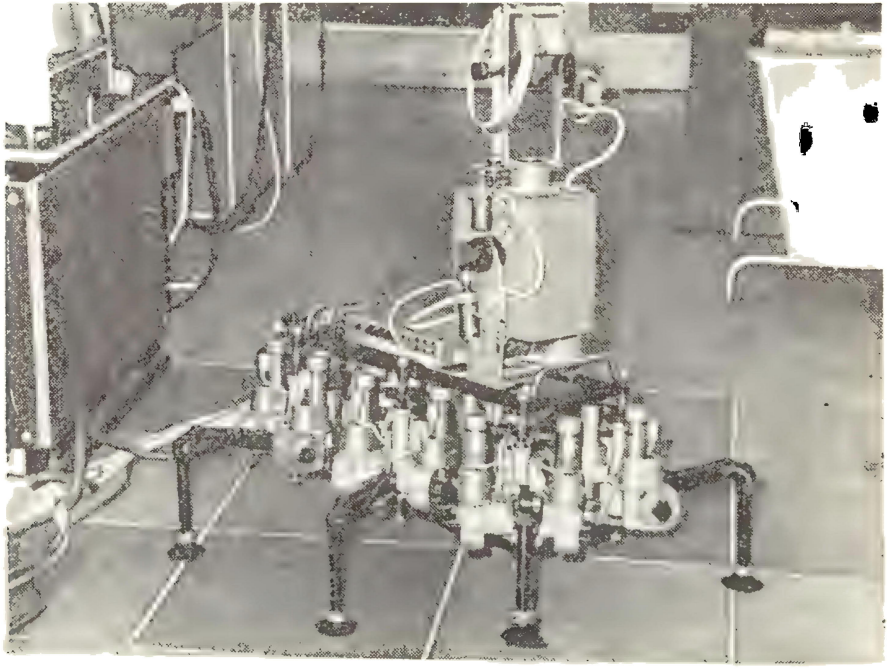


Fig. 9.16 Six-legged digital controlled walker

This research led to a construction of the six-legged machines capable of carrying necessary equipment and possessing means of sensory perception. These models (Figs 9.15 and 9.16) connected to digital and analog control systems helped elaborate the composition and structure of a control system of a walking robot capable of moving over uneven terrain in an automatic or supervisory mode [10, 62].

9.5 RESULTS

This research allows us to formulate a number of statements on the principles of elaborating a control system of walking vehicles. The motion activity at all levels should be planned using a certain set of standard actions developed in advance and introduced to the robot memory. Thus, instead of the known self-teaching concept we put forward a more effective concept of adjusting the robot to an assigned motion activity.

Then, planning involves selecting a sequence of standard motions (depending on the environment and aims of the motion) and their slight modifying by choosing the values of free parameters. The method applied at several planning levels produces a rich motor behavior and requires small computer resources.

With reference to the legged machine, the set of standard actions complies with earlier mentioned Bernstein's postulate stating that a reduction of redundant degrees of freedom is achieved by motion coordination. This can be implemented in several ways.

The most simple way is to fix some degrees of freedom. This method is often used by crane and excavator operators to simplify the control. As laboratory experiments and mathematical modeling of a specified kinematic system of a legged vehicle have demonstrated, it is permissible in a number of cases to fix knee joints of the support legs and femur and knee joints of the transfer legs. It leads to an inconsiderable violation of the motion pattern but significantly reduces the control processor loading.

A more complicated method is to use motion synergies determining a fixed motion pattern. A concept of synergies is a concept of modes of solving motion problems worked out in teaching a robot. Each mode represents a definite sequence of actions and a combination of kinematic links in joints. So to perform motions it is enough to control a small number of independent parameters though the number of degrees of freedom participating in the motion can be greater.

Regular wave gaits of a walker may serve a typical example. With this mode the motion of each leg is controlled by a special leg-stroke generator assigning the motion laws to the leg joints, and the leg joint motion is controlled by a system of interlimb coordination [12]. The leg-stroke generator provides a change of the leg motion phases—lifting, transferring, placing, and supporting. The sy-

stem of interlimb coordination governs a sequence of actuating leg-stroke generators depending on the gait and motion feedback. The method was realized by devising dedicated analog hardware.

Digital software/hardware means ensured realization of a third method: a set of standard actions is transformed into a package of subroutines controlled by a system monitor program. The subroutines are called in by the environment conditions and by the operation results of the subroutines called in before. Obviously the robot control system is adjusted by forming elementary executing programs and rules governing the sequence of elementary actions in standard and nonstandard situations.

An illustrative example is furnished by a set of elementary actions of planning, comprising predicting a body motion with reference to the ground, selecting a gait and path, forming a footprint pattern, and planning a motion law in the leg joints. This set is realized in a library of subprograms the call of which and data transfer are controlled by stack memory and information files. The monitor of the motion constructing system controls the stack memory content variation effecting the sequence of calling library programs depending on control signals fed to the monitor by the executors through internal interruption means or through information files [13, 39, 59, 60, 61].

Planning must be carried out in close cooperation with the information system the characteristics of which completely determine the composition and content of the set of standard actions of motion construction. As the experience has shown, the data perception cannot be separated from the aims of motion, which often considerably simplifies the information system by making pattern recognition unnecessary, and thus converting it to a simple detector of obstacles.

The interaction of the information system and the system of motion construction leads to procedures of active surveillance.

Let us discuss as an example the rules of active surveillance in the control system of a walking vehicle fitted with a location range finder:

—in a regular case the center of the scanning sector follows the point moving along the planned path at a certain distance from the robot;

—if the information is insufficient the system of motion construction requests the scanning sector to go up to increase the distance of surveillance;

—if the path was earlier screened by obstacles, the scanning sector goes down for revision and then returns to the initial position;

—in those cases when the range is too great the scanning sector goes down;

—in the case of an insurmountable obstacle the surveillance sector turns from the extreme right to the extreme left position.

The decision making is to be organized in the form of the glossary "situation-action" (or "situation-object"), which establishes correlation between the results of the sensor signals analysis and that motion act (or a signal of the percepted object), which corresponds to the situation. Glossaries "situation-action" simplify a logic structure of the algorithm, allow specialized hardware to be used, and facilitate robot adjustment to solving an assigned problem [34].

Due to a finite resolving capacity of the receptors the dimension of the situation space is not large, which facilitates the elaboration of such glossaries. For a mobile robot a natural unit of measurement in its "world" is a characteristic dimension of its body. The length of the legs, step size, region of surveillance are expressed in this measurement.

As the experience has shown, the robot's world does not exceed 10 robot bodies and is not less than 1% of the body length (foot size). These are limiting values, which determine the space quantum (i.e. a required precision of the information system) and the volume of information in the robot's world (i.e. the requirements to the storage capacity of the control computer).

The use of a location range finder to form a geometric model of environment has shown that algorithms of perception and motion planning essentially depend on the relation between the speed and distance of the surveillance, and the locomotion speed. It is desirable that the time of surveying within the surveillance range be a small value with reference to the time of covering this range.

References

1. БЕРНШТЕЙН Н. А.: *О построении движения.* Медгиз, 1947.

2. ВАСЕНИН В. А., ДЕВЯНИН Е. А., ЖИХАРЕВ Д. Н. и др.: Макет шагающего аппарата и его система управления. *Изв. АН СССР, Техническая кибернетика,* 6 (1974).

3. ВЕРЕЩАГИН А. Ф., ГЕНЕРОЗОВ В. Л., КУЧЕРОВ В. Б.: Алгоритмы управления манипуляторов по вектору скорости. *Изв. АН СССР, Техническая кибернетика,* 3 (1975).

4. ГАНТМАХЕР Ф. Р.: *Теория матриц.* Наука, Москва, 1966, 576с.

5. GASS S. I.: *Linear programming. Methods and Applications.* McGraw-Hill, New York, 1958.

6. ГЕРХЕН-ГУБАНОВ Г. В., КУЗНЕЦОВ В. Г.: Дальномерная обзорно-информационная система для распознавания трехмерных предметов. В кн.: *IV Всесоюзн. симпоз. по теории и принципам роботов и манипуляторов.* Секция П., Тольятти, 1976, с. 40.

7. ГРЕЧАНОВСКИЙ Е. Н., ПИНСКЕР И. Ш.: Метод планирования движений манипулятора при наличии препятствий. В сб. *Модели. Алгоритмы. Принятие решения.* Наука, Москва, 1979.

8. ГУРФИНКЕЛЬ В. С., ДЕВЯНИН Е. А., ГУРФИНКЕЛЬ Е. В. и др.: Организация локомоторного цикла шагающего аппарата. В кн.: *Материалы IV Всесоюзн. совещ. по пробл. управления.* Наука, Москва, 1974.

9. ГУРФИНКЕЛЬ В. С., ШНЕЙДЕР А. Ю., КАНАЕВ Е. М., ОСТАПЧУК В. Г.: Модель шагающего аппарата с автономным управлением. В кн.: *Бионика.* Наука, Москва, VI, 1973.

10. ДЕВЯНИН Е. А., ГУРФИНКЕЛЬ В. С. и др.: Адаптивный робот с регулируемой жесткостью. В кн.: *Материалы II Всесоюзн. совещ. по робототехнич. системам.* Минск, 1981, ч. III, 81-83.

11. ДЕВЯНИН Е. А., ЛЕНСКИЙ А. В. и др.: Макет управляемого шагающего аппарата. В кн.: *Бионика.* Наука, Москва, 1973, т. VI.

12. ДЕВЯНИН Е. А., ЛЕНСКИЙ А. В., САМСОНОВ В. А.: Задача управления движением шагающего аппарата. В кн.: *Биомеханика,* Рига, 1975 (Тр. РНИТО; вып. 13).

13. ДЕВЯНИН Е. А., САМСОНОВ В. А., ШТИЛЬМАН Л. Г.: Обработка макета шагающего аппарата и его систем управления. В кн.: *Управление в пространстве.* Наука, Москва, 1975, т. 2.

14. DUDA R. and HARDT P. E.: *Pattern Classification and Scene Analysis.* Wiley & Sons, New York-London-Sidney-Toronto, 1973.

15. ЗУХОВИЦКИЙ С. Н., АВДЕЕВА Л. И.: *Линейное и выпуклое программирование.* Наука, Москва, 1967, 460 с.

16. ИВАХНЕНКО А. Г.: *Системы эвристической самоорганизации в технической кибернетике.* Техника, Киев, 1971, 372 с.

17. ИВКИН А. М., КУДРЯШОВ В. Б., ЮЩЕНКО А. С.: Применение принципа самонаведения для управления манипулятором. *Изв. АН СССР, Техническая кибернетика*, 6 (1979).

13. ИГНАТЬЕВ М. Б., КУЛАКОВ Ф. М., ПОКРОВСКИЙ А. М.: *Алгоритмы управления роботами-манипуляторами.* Машиностроение, Ленинград, 1977, 248 с.

19. ИНОУЭ Х. Операционная система для управления движением манипулятором робота ЭТЛ. В сб.: Интегральные роботы, т. 2, Москва, Мир, 1975, 323 с.

20. КОБРИНСКИЙ А. А., КОБРИНСКИЙ А. Е.: Построение движений манипуляционной системы в среде с препятствиями. *ДАН СССР*, 224, 6 (1975).

21. КУЛАКОВ Ф. М., НЕОКЕСАРИЙСКИЙ В. Н.: Математическая модель эффекторной системы робота. *Препринт № 004, ЛНИВЦ АН СССР*, Ленинград, 1980.

22. КУЛАКОВ Ф. М.: *Супервизорное управление манипуляционными роботами.* Наука, Москва, 1980, 448 с.

23. КУЛЕШОВ В. С., ЛАКОТА Н. А.: *Динамика систем управления манипуляторами.* Энергия, Москва, 1971, 304 с.

24. КУЛЕШОВ В. С., ЮЩЕНКО А. С.: Некоторые алгоритмы позиционного управления манипуляторами. *Изв. АН СССР, Техническая кибернетика*, I (1977).

25. CAMPBELL-KELLY, M.: *An Introduction to Macros*, McDonald, London and American Elsevier, Inc., New York, 1973.

26. LOEFF L. A. and SONI A. H.: An Algorithm for Computer Guidance of a Manipulator in Between Obstacles. *Transactions of the ASME. Journal of Engineering for Industry*, 97, Series B, Number 3, Aug. 1975, 836-842.

27. ЛОДОЧНИКОВ Э. А., ЮФЕРОВ Ф. М.: *Микроэлектродвигатели для систем автоматики.* Энергия, Москва, 1969.

28. Laurent P.-J.: *Approximation et optimisation*, Paris, 1972.

29. ЛУРЬЕ А. И.: *Аналитическая механика.* Физматгиз, Москва, 1961. 824 с.

30. МАКАРОВ И. М., РАХМАНКУЛОВ В. З.: Групповое управление роботами-манипуляторами с распределенно-централизованной организацией обработки информации. Сб. *Будущее науки.* Выпуск 14, Знание, 1981.

31. МАКАРОВ И. М.: Робототехника: проблемы и перспективы. Сб. *Микропроцессорные системы управления в робототехнике.* Наука, Москва, 1982.

32. МЕДВЕДЕВ В. С., ЛЕСКОВ А. Г., ЮЩЕНКО А. С.: *Системы управления манипуляционных роботов.* Наука, Москва, 1978, 415 с.

33. NEWMAN W., SPROULL R.: *Principles of Interactive Computer Graphics*, New York, McGraw-Hill, 1973.

34. ОХОЦИМСКИЙ Д. Е., ПЛАТОНОВ А. К., БОРОВИН Г. К. и др.: Алгоритмы управления движением шагающего аппарата. *Препринт ИПМ АН СССР*, 63, Москва, 1972.

35. ОХОЦИМСКИЙ Д. Е., ПЛАТОНОВ А. К., БОРОВИН Г. К., КАРПОВ И.: Моделирование на ЦВМ движения шагающего аппарата. *Препринт*

ИПМ АН СССР, 1971.— *Изв. АН СССР, Техническая кибернетика*, 1972, вып. 3.

36. ОХОЦИМСКИЙ Д. Е., ПЛАТОНОВ А. К., ГУРФИНКЕЛЬ В. С., ДЕВЯНИН Е. А.: Моделирование локомоционных систем. В кн.: *Труды IV Всесоюзн. конф. по бионике.* Москва, 1973, т. I.

37. ОХОЦИМСКИЙ Д. Е., ПЛАТОНОВ А. К. и др.: Координация движений шагающего аппарата. В кн.: *Бионика*, 1973, т. VI, 176—181.

38. ОХОЦИМСКИЙ Д. Е., ПЛАТОНОВ А. К. и др.: Управление интегральным локомоционным роботом. *Изв. АН СССР, Техническая кибернетика*, 6, (1974).

39. ОХОЦИМСКИЙ Д. Е., ПЛАТОНОВ А. К., КУГУШЕВ К. И. и др.: Проблемы построения и моделирования движения управляемого оператором шагающего аппарата. *Препринт ИПМ АН СССР*, 125, Москва, 1974.

40. PARTHASARATHY T., RAGHAVAN T., *Some Topics in Two-Person Games.* American Elsevier Publishing Company, New York, 1971.

41. ПЛАТОНОВ А. К., ПРЯНИЧНИКОВ В. Е.: Моделирование внешней среды локомоционного робота. *Препринт ИПМ АН АСССР*, 84, Москва, 1977.

42. PAUL R.: Modelling Trajectory Calculation and Serving of a Computer Controlled Arm. *Stanford Artificial Intelligence Project. Memo AIM*-177. *Computer Science Department Report CS*-311, November 1972.

43. ПОПОВ Е. П., ВЕРЕЩАГИН А. Ф., ЗЕНКЕВИЧ С. Л.: *Манипуляционные работы. Динамика и алгоритмы.* Наука, Москва, 1978, 400 с.

44. *Проектирование следящих систем двустороннего действия.* Под ред. Кулешова В. С. Машиностроение, Москва, 1980, 300 с.

45. ROCKEFELLER R. T.: *Convex analysis*, Princeton Universitv Press, 1970.

46. СОЛОДОВНИКОВ А. С.: *Системы линейных неравенств.* Наука, Москва, 1977.

47. СУББОТИН Ю. Н., СТЕЧКИН С. Б.: *Сплайны в вычислительной математике.* Наука, Москва, 1976, 248 с.

48. SAGE A. P. and MELSE J. L.: Estimation Theory with Application to Communication and Control, New York, 1972.

49. ТОПЧЕЕВ Ю. И., ЧЕМБРОВСКИЙ, САМОЙЛОВИЧ. *Общие принципы проектирования ракетных комплексов.* Машиностроение, Москва.

50. WHITNEY D. E.: The Mathematics of Coordinated Control of Prosphetic Arms and Manipulators. *Transactions of ASME, Journal of Dynamic Systems, Measurement, and Control*, 94, December 1972, 303-309.

51. HALFMAN R. L.: *Dynamics.* Addison-Wesley Publishing Company, Inc. Reading, Massachusetts, Palo Alto, London.

52. Helms D.: Entwicklung eines Seckskomponenten Kraft-Moment-Aufnemers. *Feinwerktechnik und Messtechnik*, 85 (1977) 2, 75-81.

53. ЮРЕВИЧ Е. И. (ред.). *Управление роботами от ЭВМ.* Энергия, Ленинград, 1980, 261 с.

54. ЮЩЕНКО А. С.: Исследование малых колебаний манипулятора, управляемого с помощью ЭВМ. *Известия ВУЗов СССР*, 11 (1980) Машиностроение, Москва.

55. ЮЩЕНКО А. С. Комбинированная система полуавтоматического управления манипулятором. *Изв. АН СССР, Техническая кибернетика*, 1 (1979).

56. ЮЩЕНКО А. С.: О взаимосвязи алгоритмов управления манипуляторами по вектору силы и по вектору скорости. Сб. *Управление роботами и манипуляторами*. Наука, Москва, 1981.

57. ЮЩЕНКО А. С., ШИГОРИН Л. П.: О применении метода регуляризации для решения задачи об управлении манипуляторами. *Изв. ВУЗов СССР*, 6 (1977), Машиностроение, Москва.

58. KUGUSHEV E. I., JAROSHEVSKY V. S.: Problems of selecting a gait. *Proc. IV Intern. Joint. Conf. Artificial Intelligence*, Tbilisi, USSR, 1975, Sept. 3-8.

59. OKHOTSIMSKI D. E., PLATONOV A. K.: Control algorithm of the walker climbing over obstacles. In: *Third Intern. Joint Conf. Artificial Intelligence*. Stanford (Cal.), 1973.

60. OKHOTSIMSKI D. E., PLATONOV A. K.: On Problems and principles of robot motion. In: *Proc. 14th IUTAM Congr.*, Delft, 1976.

61. OKHOTSIMSKI D. E., PLATONOV A. K.: Perceptive Robot Moving in 3-d World. In: *Proc. IV Intern. Joint Conf. Artificial Intelligence*, Tbilisi, USSR, 1975, Sept. 3-8.

62. OKHOTSIMSKI D. E., PLATONOV A. K.: Walker's motion control. In: *Proc. 2nd CISM-IFTOMM Symp. «ROMANSY»*, Warsaw, 1976.

63. WILSON D. M.: Insect Walking, *Annual Review of Entomology*, 11, 1966, 103-122.

Index

A

Actuator control,
  block diagram of, 79
  frequency response of, 72
  kinematic diagram of, 80
Actuator system,
  block diagram of, 63
  damping factors of, 64
  dynamic precision for, 76
  multivariable, 64
  natural modes of, 64
  synthesis of, 62
Algorithm,
  automatic force control, 163
  combined force control, 162
  combined rate control, 155
  force control, 161
  free gait, 178
  integral robot control, 181
  kinematic control, 150, 163
  locomotion, uneven relief, 183
  locomotion, with prismatic obstacles, 181
  motion, 178
  position control, 170
  rate control, 150, 169, 170
  semiautomatic control, 150
Automatic feeder, 15

C

Carrier inertia tensor, 50
Communication channels,
  testing of, 135
Computer graphics, 136
Control systems,
  semiautomatic, 149-175
Coordinate systems, 23
  base-tied, 25
  hand-tied, 24
  transition illustrated for, 25, 27

D

Distance subdifferentials, 86

E

End effector, 24
Equations of motion,
  dynamic, 42
  kinematic, 31
  linearizing of, 51
  moving-base robot, dynamic, 48

F

Flow line, 13
  digraph of a robot-served, 17
  model of, 13
  queueing systems for, 15
  with R manipulators, 16

G

Gait, 176
  free, 179
  insects wave, 179
  regular, 179
  symmetric, 179

Generalized coordinates, 28
Generalized coordinate domain, 109

H

Hand state, 83

I

Inf-convolution, 88

K

Kinematics,
  basic equation of, 31
  partitioned matrix representation, 37

L

Link,
  absolute angular acceleration of, 34
  absolute angular velocity of, 32
  position with respect to a body, 91
Linkage, 23
  anthropomorphic robot, 114
  configurations of, 97
  dynamic equation of, 42
  model of, 82
  path synthesis for, 82
Master-slave manipulator systems, 156
Model of flow line, 13
Motion control,
  correction, 177
  language, 126
  software, 131
Motion planning, 176
Moving-base robot, 48
  dynamic equations for, 48
  frames tied to, 50

N

Nyquist criterion, 67, 68

O

Operating system,
  design of, 116
  robot-oriented, 118
  universal, 118

P

Pairs,
  revolute, 23, 37, 47
  screw, 30, 37, 46, 47
  sliding, 23, 30, 47
Path synthesis, 82
  by minimizing a functional, 109
  Chebyshev distance for, 84
  distance evaluation in, 86
  Euclidean distance for, 84, 86
  LP for, 84
  software for, 90
Position vector domain, 106
Programmed path,
  extrapolating of, 112

R

Robot carrier, 51
Robot control,
  approaches to, 105
  automatic, 171
  dynamic, 159
  force, 171
  hand self-directing, 157, 159
  in assembly operations, 100
  kinematic, 150
  model of, 100
  precise position, 156
  rate, 150, 172
  semiautomatic, 150, 165, 172, 173
  supervisory, 181
  walking motion, 176-191
Robot dynamics,
  mathematical model of, 23
Robotic complex,
  design of, 116

S

Software,
game theory analysis for, 144-148
methods for evaluation, 139-148
motion control, 118, 131-133
pattern recognition technique for, 139-144
robot-oriented, 116-138
robot teaching, 118-133, 134
Spectral frequency response analysis, 65
Synthesis of robot actuators, 75

T

Transition matrix in flow line analysis, 18

W

Walking vehicle, 176
mathematical modeling of, 178, 184
physical modeling of, 184
ricksha-type model of, 186, 187

TO THE READER

Mir Publishers welcome your comments on the content, translation, and design of the book. We would also be pleased to receive any suggestions you care to make about our future publications.

Our address is:
USSR, 129820,
Moscow, I-110, GSP,
Pervy Rizhsky Pereulok, 2

Also in this series:

MOLECULAR GAS LASERS —
PHYSICS AND APPLICATIONS

Edited by E. Velikhov

This is a collection of papers by leading Soviet authorities on laser physics and engineering. Among the topics discussed in the book are the physics of gas discharges, elementary processes in lasers, plasma chemistry, and the kinetics of laser media. Several excitation methods are proposed for molecular lasers, including eximeric and chemical. The engineering aspects concern the use of lasers for isotope separation and in laser thermonuclear reactions. Intended for specialists concerned with laser engineering and applications.

Modern Robot Engineering

Edited by E. Popov

This is a collection of papers by leading Soviet authorities on robotics cemented by a unified approach to the problem of robot kinematics, path synthesis, and control. The papers consider the selection of structure and parameters for robotic systems and robot-served flow lines, analyze the dynamics and kinematics of robot arm linkage, and synthesize paths in crowded workspace. Among the topics of the book are programs specifically devised to control and teach robots, along with the comparison of these programs for performance.

The presented theory is discussed as applied to built-in robotic systems, robots with various principles of control, and walking vehicles.